

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-050554

(43)Date of publication of application : 20.02.1996

(51)Int.Cl. G06F 11/22
G06F 9/38

(21)Application number : 07-060487

(71)Applicant : FUJITSU LTD

(22)Date of filing : 20.03.1995

(72)Inventor : IWASHITA HIROAKI
FURUWATARI SATOSHI
NAKADA TSUNEO
HIROSE FUMIYASU

(30)Priority

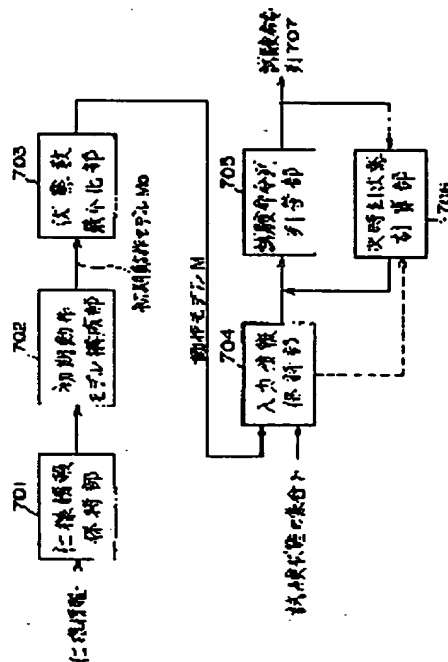
Priority number : 06116705 Priority date : 30.05.1994 Priority country : JP

(54) METHOD AND DEVICE FOR AUTOMATICALLY GENERATING OPERATION MODEL OF PROCESSOR AND TEST INSTRUCTION SEQUENCE FOR LOGIC VERIFICATION

(57)Abstract:

PURPOSE: To automatically generate a high-quality test instruction sequence in a short time for verifying the operation logic of a processor equipped with a pipelined processing function.

CONSTITUTION: A part 701 holds specification information concerning the pipelined configuration of the processor and an instruction to be executed by the processor. A part 702 constitutes an initial operation model M0 concerning a pipeline from the specification information. A part 703 constitutes an operation model M by minimizing the number of states of the initial operation model M0. A part 705 successively presents a test instruction sequence 707 corresponding to the process to transit from a prescribed input state to any test state contained in a set H of test states without competing the state of the operation model M based on the operation model M held in the part 704 and the set H of test states. A part 706 calculates the next time state of a state to be next transit after the state of the operation model M is turned to the test state corresponding to the test instruction sequence 707, and that next time state is inputted to a successive test instruction sequence presenting means 705 as the new input state.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision
of rejection]

[Date of requesting appeal against examiner's
decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2000 Japanese Patent Office

(51) Int.Cl. ⁴	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 11/22	3 1 0 U			
9/38	3 8 0 C			

審査請求 未請求 請求項の数18 O L (全 46 頁)

(21) 出願番号 特願平7-60487

(22) 出願日 平成7年(1995)3月20日

(31) 優先権主張番号 特願平6-116705

(32) 優先日 平6(1994)5月30日

(33) 優先権主張国 日本 (J P)

(71) 出願人 000005223

富士通株式会社

神奈川県川崎市中原区上小田中1015番地

(72) 発明者 岩下 洋哲

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(72) 発明者 古渡 聡

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(72) 発明者 中田 恒夫

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(74) 代理人 弁理士 大菅 義之 (外1名)

最終頁に続く

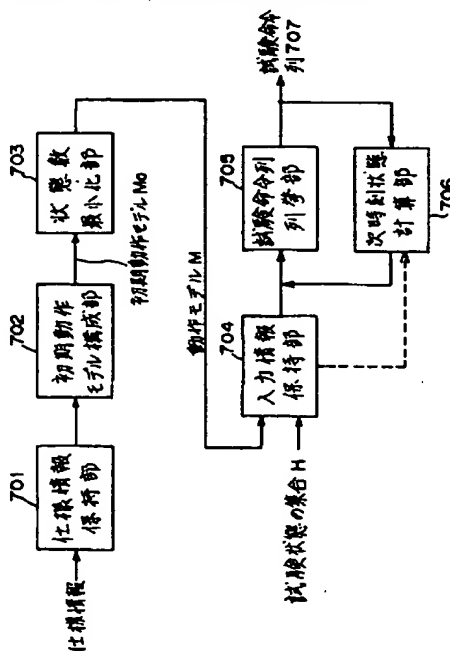
(54) 【発明の名称】 プロセッサの動作モデルと論理検証用試験命令列の自動生成方法及び装置

(57) 【要約】

【目的】 パイプライン処理機能を有するプロセッサの動作論理を検証するため質の高い試験命令列を自動的にかつ短時間に生成可能とすることを目的とする。

【構成】 701の部分は、プロセッサのパイプライン構成及びプロセッサが実行する命令に関する仕様情報を保持する。702の部分は、仕様情報から、パイプラインに関する初期動作モデルM₀を構成する。703の部分は、初期動作モデルM₀の状態の数を最小化することにより、動作モデルMを構成する。705の部分は、704の部分に保持された動作モデルMと試験状態の集合Hとから、動作モデルMの状態が競合を起こすことなく所定の入力状態から試験状態の集合Hに含まれる試験状態のうちの何れかへ遷移する過程に対応した試験命令列707を列挙する。706の部分は、動作モデルMの状態が試験命令列707に対応する試験状態になった後に次に遷移する状態である次時刻状態を計算し、その次時刻状態を新たな入力状態として試験命令列列挙手段705に入力させる。

本発明の第1の実施例の全体構成図



【特許請求の範囲】

【請求項 1】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデルを構成し、該動作モデルに含まれる複数の状態変数の状態が競合を起こすような状態である試験状態を列挙し、前記動作モデルの状態がその初期状態から前記各試験状態へ遷移する過程に対応した試験命令列を生成する、ことを特徴とするプロセッサの動作モデルと論理検証用試験命令列の自動生成方法。

【請求項 2】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報を保持する仕様情報保持手段と、該仕様情報保持手段が保持する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデルを構成する動作モデル構成手段と、該動作モデル構成手段によって構成される動作モデルに含まれる複数の状態変数の状態が競合を起こすような状態である試験状態を列挙する試験状態列挙手段と、前記動作モデル構成手段によって構成される動作モデルの状態がその初期状態から前記試験状態列挙手段によって列挙される各試験状態へ遷移する過程に対応した試験命令列を生成する試験命令列生成手段と、を有することを特徴とするプロセッサの動作モデルと論理検証用試験命令列の自動生成装置。

【請求項 3】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデルを初期動作モデルとして構成し、該初期動作モデルに含まれる各状態変数が有する複数の状態を、それらの状態が有する属性に基づいてグループ化することにより、前記各状態変数が有する状態の数を最小化し、該最小化が行われた動作モデルを構成する、ことを特徴とするプロセッサの動作モデルの自動生成方法。

【請求項 4】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報を保持する仕様情報保持手段と、該仕様情報保持手段が保持する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデルを初期動作モデルとして構成する初期動作モデル構成手段と、該初期動作モデル構成手段によって構成される初期動作

モデルに含まれる各状態変数が有する複数の状態を、それらの状態が有する属性に基づいてグループ化することにより、前記各状態変数が有する状態の数を最小化し、該最小化が行われた動作モデルを構成する状態数最小化手段と、

を有することを特徴とするプロセッサの動作モデルの自動生成装置。

【請求項 5】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態が競合を起こすような状態である試験状態を、入力情報として入力し、該入力情報として入力される動作モデルを構成する複数の状態変数の状態が競合を起こす状態を含むことなく、前記動作モデルの状態が所定の入力状態から前記入力情報保持手段に保持される試験状態のうちの何れかへ遷移する過程に対応した試験命令列を列挙し、前記入力情報として入力される動作モデルを構成する複数の状態変数の状態が、前記列挙された試験命令列に対応する試験状態になった後に次に遷移する状態である次時刻状態を計算し、該次時刻状態を前記所定の入力状態として前記試験命令列を列挙する処理を繰り返し実行させる、ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項 6】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態が競合を起こすような状態である試験状態を、入力情報として保持する入力情報保持手段と、該入力情報保持手段に保持された動作モデルを構成する複数の状態変数の状態が競合を起こす状態を含むことなく、前記動作モデルの状態が所定の入力状態から前記入力情報保持手段に保持される試験状態のうちの何れかへ遷移する過程に対応した試験命令列を列挙する試験命令列列挙手段と、

前記入力情報保持手段に保持された動作モデルを構成する複数の状態変数の状態が、前記試験命令列列挙手段が列挙した試験命令列に対応する試験状態になった後に次に遷移する状態である次時刻状態を計算し、該次時刻状態を前記所定の入力状態として前記試験命令列列挙手段に入力させる次時刻状態計算手段と、

を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項 7】 プロセッサのパイプラインの構成及び前

記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態を入力とする状態評価関数を、入力情報として入力し、

該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態であって、前記状態評価関数の値が所定の条件を満たす状態を列挙し、

該列挙される状態に至る命令列を試験命令列として列挙する、

ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項 8】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態を入力とする状態評価関数を、入力情報として保持する入力情報保持手段と、

該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態であって、前記状態評価関数の値が所定の条件を満たす状態を列挙する到達可能状態列挙手段と、

該到達可能状態列挙手段によって列挙される状態に至る命令列を試験命令列として列挙する試験命令列列挙手段と、

を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項 9】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態を入力とする状態評価関数を、入力情報として入力し、

該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態を列挙し、

該列挙される状態に対応する前記状態評価関数の値を計算し、

前記列挙される状態に至る命令列を列挙し、

前記計算される値が所定の条件を満たす状態を前記列挙される状態から選択し、

該選択された状態に対応する命令列を前記列挙される命令列から選択し、

該選択された命令列を試験命令列として選択する、

ことを特徴とする論理検証用試験命令列の自動生成方

法。

【請求項 10】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態を入力とする状態評価関数を、入力情報として保持する入力情報保持手段と、

該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態を列挙する到達可能状態列挙手段と、

該到達可能状態列挙手段によって列挙される状態に対応する前記状態評価関数の値を計算する状態評価関数計算手段と、

前記到達可能状態列挙手段によって列挙される状態に至る命令列を列挙する命令列列挙手段と、

前記状態評価関数計算手段によって計算される値が所定の条件を満たす状態を前記到達可能状態列挙手段によって列挙される状態から選択し、該選択された状態に対応する命令列を前記命令列列挙手段から選択し、該選択された命令列を試験命令列として出力する試験命令列出力手段と、

を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項 11】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列を、入力情報として入力し、

前記入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、前記入力情報として入力された状態の時系列を列挙し、

該列挙される状態の時系列を実現する命令列を試験命令列として列挙する、

ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項 12】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列を、入力情報として保持する入力情報保持手段と、

前記入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、前記入力情報として入力される状態

の時系列を列挙する到達可能状態時系列列挙手段と、
該到達可能状態時系列列挙手段によって列挙される状態の時系列を実現する命令列を試験命令列として列挙する試験命令列列挙手段と、
を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項 1 3】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列を、入力情報として入力し、
該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態を列挙し、
前記列挙される状態に至る命令列を列挙し、
前記列挙された状態からバックトラックしながら、前記入力情報として入力される状態の時系列に一致する状態の時系列を検索し、
該検索された状態の時系列の最終状態に至る命令列を前記列挙された命令列から選択し、
該選択された命令列を試験命令列として出力する、
ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項 1 4】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列を、入力情報として保持する入力情報保持手段と、
該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態を列挙する到達可能状態列挙手段と、
前記到達可能状態列挙手段によって列挙される状態に至る命令列を列挙する命令列列挙手段と、
前記到達可能状態列挙手段によって列挙された状態からバックトラックしながら、前記入力情報として入力される状態の時系列に一致する状態の時系列を検索する状態時系列検索手段と、
該状態時系列検索手段によって検索された状態の時系列の最終状態に至る命令列を前記命令列列挙手段から選択し、該選択された命令列を試験命令列として出力する試験命令列出力手段と、
を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項 1 5】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列の評価関数を、入力情報として入力し、
前記入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、前記評価関数の値が所定の条件を満たす状態の時系列を列挙する到達可能状態時系列列挙手段と、
該到達可能状態時系列列挙手段によって列挙される状態の時系列を実現する命令列を試験命令列として列挙する試験命令列列挙手段と、
を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項 1 6】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列の評価関数を、入力情報として保持する入力情報保持手段と、
前記入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、前記評価関数の値が所定の条件を満たす状態の時系列を列挙する到達可能状態時系列列挙手段と、
該到達可能状態時系列列挙手段によって列挙される状態の時系列を実現する命令列を試験命令列として列挙する試験命令列列挙手段と、
を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項 1 7】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列の評価関数を、入力情報として入力し、
該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態の時系列について、前記評価関数を計算し、
該計算される値が所定の条件を満たす状態の時系列を前記列挙された状態からバックトラックして得られる状態の時系列のうちから選択し、
該選択された状態の時系列の最終状態に至る命令列を前記列挙された命令列から選択し、
該選択された命令列を試験命令列として出力する、
ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項 18】 プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列の評価関数を、入力情報として保持する入力情報保持手段と、該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態を列挙する到達可能状態列挙手段と、前記到達可能状態列挙手段によって列挙される状態に至る命令列を列挙する命令列列挙手段と、前記到達可能状態列挙手段によって列挙された状態からバックトラックして得られる状態の時系列について、前記評価関数を計算する状態時系列評価関数計算手段と、該状態時系列評価関数計算手段によって計算される値が所定の条件を満たす状態の時系列を前記到達可能状態列挙手段によって列挙された状態からバックトラックして得られる状態の時系列のうちから選択し、該選択された状態の時系列の最終状態に至る命令列を前記命令列列挙手段から選択し、該選択された命令列を試験命令列として出力する試験命令列出力手段と、を有することを特徴とする論理検証用試験命令列の自動生成装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、パイプライン処理機能を有するプロセッサの動作論理を検証するための命令列を自動生成する技術に関する。

【0002】

【従来の技術】プロセッサの機能又は構造の設計が仕様を正しく実現しているか否かを検証するための論理シミュレーション、及び製造されたプロセッサが仕様通りに動作するか否かを検証するための動作試験を行うためには、プロセッサに与えられる試験命令列（テストプログラム）の作成が不可欠である。更に、短時間で信頼性の高い検証を行うためには、質の良いテストプログラムを作成する必要がある。

【0003】ここで、プロセッサの内部で命令のパイプライン処理を行うことは、処理速度の高いプロセッサを実現するために欠かせない技法である。パイプライン処理が実行される場合、同時に処理される複数の命令の間で、次のような競合状態（パイプラインハザード）が発生する可能性がある。

【0004】・ある機能を有するハードウェア資源の数がNである場合に、合計でN+1以上の当該ハードウェア資源の使用要求が発生する状態。

・2つの命令間のレジスタ又はメモリに対するデータの読出し及び書込みの順序が入れ替わる結果、正常な実行結果が得られなくなる状態。

これらのパイプラインハザードを解決するためには、プロセッサ内に、パイプラインのインターロック及び命令の実行順序の入換え等を制御する機構が必要である。

【0005】このような機構を実現するためのプロセッサの機能又は構造の設計が仕様を正しく実現しているか否かを検証するためには、論理シミュレーションを実行する必要がある。更に、製造されたプロセッサが仕様通りに動作するか否かを検証するために、動作試験を行う必要がある。そして、これらの検証を行うためには、プロセッサに与える試験命令列（テストプログラム）の作成が不可欠となる。

【0006】プロセッサの構造は、一般に、実際のデータ処理を扱う演算部（ALU、レジスタファイル等）と、それらを制御する制御部に分けることができる。そして、そのような構造のうちでも、制御部が有する機能は複雑である。そのため、その機能を効率よく検証できるテストプログラムを作成することは、重要であると同時に難しい問題である。

【0007】従来のテストプログラムは、次のような方法の組み合わせによって作成されている。

方法1：試験命令列を手で作成する。

【0008】方法2：計算機を利用して、無作為に多くの試験命令列を発生させる。

方法3：既存のプログラムを利用して試験命令列を作成する。

なお、プロセッサの仕様を表現する記述からプロセッサの制御機構を検証するための試験命令列を自動生成する方式に関しては、本願出願人により既に2件の特許出願：特願平4-145812、特願平5-217813がなされているが、これらは本発明が実現しているような厳密な動作モデルに基づいたものではない。

【0009】

【発明が解決しようとする課題】しかし、上述した方法1に基づく従来技術は、人手を利用するため、多くの開発期間を必要とする。更に、人手で作成された試験命令列は、その質を評価することが難しく、その有効性を客観的に保証することができないという問題点を有している。

【0010】また、上述した方法2又は3に基づく従来技術では、命令数を増やすことによって検証の信頼性を上げることが可能ではある。しかし、論理シミュレーションや動作試験を行うためには、膨大な実行時間が必要となるという問題点を有している。

【0011】更に、本願出願人により既に特許出願されている方式は、パイプラインハザードの発生時に前段側のパイプラインステージが停止するような制御を検証するための試験命令列の生成方式に限定されるものである。従って、これらの方式は、現在多く出現しつつある複雑なパイプライン制御方式に対応しきれないという問題点を有している。

【0012】本発明は、より多くの種類のプロセッサを対象として、対象となるプロセッサのための論理シミュレーション及び動作試験の実行時間の短縮と、それらの検証の信頼性の向上を実現するために、プロセッサの動作論理を検証するための質の高い試験命令列を自動的にかつ短時間に生成可能とすることを目的とする。

【0013】

【課題を解決するための手段】図1は、本発明の第1の態様のブロック図である。まず、仕様情報保持手段102は、プロセッサのパイプラインの構成及びプロセッサが実行する各命令に関する仕様情報101を保持する。ここで、“パイプライン”とは、命令の処理を基本的に1クロックサイクル（命令サイクル）で処理できる幾つかの単位動作（パイプラインステージ）に分割し、連続する複数の入力に対する処理実行をオーバーラップさせて各パイプラインステージにおいて並列に処理する機能をいう。1つの命令を1命令サイクルの間保持することのできるハードウェア単位をパイプラインユニットとよぶ。また、“プロセッサが実行する各命令についての仕様情報”とは、プロセッサのパイプライン動作を決定する情報であって、例えば、各命令のパイプライン処理の経路や、各命令のパイプラインステージ毎のハードウェア資源の占有についての情報、及びレジスタ又はメモリ等に対するデータの読出し／書込みの実行についての情報である。

【0014】動作モデル構成手段104は、仕様情報保持手段101が保持する仕様情報101に基づき、パイプラインの動作を表現し、パイプラインの各構成部分（各パイプラインユニット）の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデル103を構成する。この動作モデル103において、各パイプラインユニットに入力する命令は、外部又は他の状態変数から1つの状態変数に入力される状態に対応付けられる。

【0015】試験状態列挙手段106は、動作モデル構成手段104によって構成される動作モデル103に含まれる複数の状態変数の状態が競合を起こすような状態である試験状態105を列挙する。

【0016】試験命令列生成手段108は、動作モデル構成手段104によって構成される動作モデル103がその初期状態から試験状態列挙手段106によって列挙される各試験状態105へ遷移する過程に対応した試験命令列107を生成する。

【0017】上述の本発明の第1の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。図2は、本発明の第2の態様のブロック図である。

【0018】仕様情報保持手段202は、プロセッサのパイプラインの構成及びプロセッサが実行する各命令に関する仕様情報201を保持する。初期動作モデル構成

手段204は、仕様情報保持手段202が保持する仕様情報201に基づいて、パイプラインの動作を表現し、パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデルを初期動作モデル203として構成する。

【0019】状態数最小化手段206は、初期動作モデル構成手段204によって構成される初期動作モデル203に含まれる各状態変数がある複数の状態を、それらの状態がある属性に基づいてグループ化することにより、各状態変数がある状態の数を最小化し、その最小化が行われた動作モデル205を構成する。

【0020】上述の本発明の第2の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。図3は、本発明の第3の態様のブロック図である。

【0021】入力情報保持手段303は、上述した本発明の第1又は第2の態様によって構成される動作モデル301（動作モデル103、205に対応する）と、その動作モデル301に含まれる複数の状態変数の状態が競合を起こすような状態である試験状態302を、入力情報として保持する。

【0022】試験命令列列挙手段306は、入力情報保持手段303に保持された動作モデル301を構成する複数の状態変数の状態が競合を起こす状態を含むことなく、動作モデル301の状態が所定の入力状態304から入力情報保持手段303に保持される試験状態302のうちの何れかへ遷移する過程に対応した試験命令列305を列挙する。

【0023】次時刻状態計算手段308は、入力情報保持手段302に保持された動作モデル301を構成する複数の状態変数の状態が、試験命令列列挙手段306が列挙した試験命令列305に対応する試験状態になった後に次に遷移する状態である次時刻状態307を計算し、その次時刻状態307を所定の入力状態304として試験命令列列挙手段306に入力させる。

【0024】上述の本発明の第3の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。図4(a)は、本発明の第4の態様のブロック図である。

【0025】入力情報保持手段403は、上述した本発明の第1又は第2の態様によって構成される動作モデル401（動作モデル103、205に対応する）と、動作モデル401に含まれる複数の状態変数の状態を入力とする状態評価関数402を、入力情報として保持する。

【0026】到達可能状態列挙手段404は、入力情報として入力される動作モデル401を構成する複数の状態変数についてその初期状態から到達可能な状態であって、状態評価関数402の値が所定の条件を満たす状態を列挙する。

【0027】試験命令列挙手段406は、到達可能状態列挙手段404によって列挙される状態に至る命令列を試験命令列405として列挙する。図4(b)は、本発明の第4の態様のより具体的なブロック図である。

【0028】入力情報保持手段409は、上述した本発明の第1又は第2の態様により構成される動作モデル407（動作モデル103、205に対応する）と、動作モデル407に含まれる複数の状態変数の状態を入力とする状態評価関数408を、入力情報として保持する。

【0029】到達可能状態列挙手段410は、入力情報として入力される動作モデル407を構成する複数の状態変数についてその初期状態から到達可能な状態を列挙する。

【0030】状態評価関数計算手段411は、到達可能状態列挙手段410によって列挙される状態に対応する状態評価関数408の値を計算する。命令列挙手段412は、到達可能状態列挙手段410によって列挙される状態に至る命令列を列挙する。

【0031】試験命令列出力手段414は、状態評価関数計算手段411によって計算される値が所定の条件を満たす状態を到達可能状態列挙手段410によって列挙される状態から選択し、選択された状態に対応する命令列を命令列挙手段412から選択し、選択された命令列を試験命令列413として出力する。

【0032】上述の本発明の第4の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。図5(a)は、本発明の第5の態様のブロック図である。

【0033】入力情報保持手段503は、上述した本発明の第1又は第2の態様によって構成される動作モデル501（動作モデル103、205に対応する）と、動作モデル501に含まれる複数の状態変数の状態の時系列502を、入力情報として保持する。

【0034】到達可能状態時系列列挙手段504は、入力情報として入力される動作モデル501を構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、入力情報として入力される状態の時系列502を列挙する。

【0035】試験命令列列挙手段506は、到達可能状態時系列列挙手段504によって列挙される状態の時系列を実現する命令列を試験命令列505として列挙する。図5(b)は、本発明の第5の態様のより具体的なブロック図である。

【0036】入力情報保持手段509は、上述した本発明の第1又は第2の態様により構成される動作モデル507（動作モデル103、205に対応する）と、動作モデル507に含まれる複数の状態変数の状態の時系列508を、入力情報として保持する。

【0037】到達可能状態列挙手段510は、入力情報として入力される動作モデル507を構成する複数の状

態変数についてその初期状態から到達可能な状態を列挙する。

【0038】命令列列挙手段511は、到達可能状態列挙手段510によって列挙される状態に至る命令列を列挙する。状態時系列検索手段512は、到達可能状態列挙手段510によって列挙された状態からバックトラックしながら、入力情報として入力される状態の時系列508に一致する状態の時系列を検索する。

【0039】試験命令列出力手段514は、状態時系列検索手段512によって検索された状態の時系列の最終状態に至る命令列を命令列列挙手段511から選択し、選択された命令列を試験命令列513として出力する。

【0040】上述の本発明の第5の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。図6(a)は、本発明の第6の態様のブロック図である。

【0041】入力情報保持手段603は、上述した本発明の第1又は第2の態様によって構成される動作モデル601（動作モデル103、205に対応する）と、動作モデル601に含まれる複数の状態変数の状態の時系列の評価関数602を、入力情報として保持する。

【0042】到達可能状態時系列列挙手段604は、入力情報として入力される動作モデル601を構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、評価関数602の値が所定の条件を満たす状態の時系列を列挙する。

【0043】試験命令列列挙手段606は、到達可能状態時系列列挙手段604によって列挙される状態の時系列を実現する命令列を試験命令列605として列挙する。図6(b)は、本発明の第6の態様のより具体的なブロック図である。

【0044】入力情報保持手段609は、上述した本発明の第1又は第2の態様により構成される動作モデル607（動作モデル103、205に対応する）と、動作モデル607に含まれる複数の状態変数の状態の時系列の評価関数608を、入力情報として保持する。

【0045】到達可能状態列挙手段610は、入力情報として入力される動作モデル607を構成する複数の状態変数についてその初期状態から到達可能な状態を列挙する。

【0046】命令列列挙手段611は、到達可能状態列挙手段610によって列挙される状態に至る命令列を列挙する。状態時系列評価関数計算手段612は、到達可能状態列挙手段610によって列挙された状態からバックトラックして得られる状態の時系列について、その評価関数608を計算する。

【0047】試験命令列出力手段614は、状態時系列評価関数計算手段612によって計算される値が所定の条件を満たす状態の時系列を到達可能状態列挙手段610によって列挙された状態からバックトラックして得ら

れる状態の時系列のうちから選択し、選択された状態の時系列の最終状態に至る命令列を命令列列挙手段 611 から選択し、選択された命令列を試験命令列 613 として出力する。

【0048】上述の本発明の第6の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。

【0049】

【作用】本発明の第1の態様では、入力された仕様情報 101 に基づきプロセッサのパイプラインの動作を表現する動作モデル 103 が自動的に構成される。この動作モデル 103 が正確かつ簡略に構成されることにより、出力される試験命令列 107 の品質向上と、その命令列に基づくプロセッサ試験の処理の効率化を実現することができる。

【0050】次に、試験状態 105 が仕様情報 101 に基づき列挙された後、初期状態から試験状態 105 への遷移過程に対応した試験命令列 107 が生成される。試験状態 105 の選び方により、目的に応じたテストプログラムを自由に作ることが可能となる。

【0051】本発明の第2の態様では、入力された仕様情報 201 に直接的に対応した動作モデルが初期動作モデル 203 として生成された後、その動作モデルに含まれる各状態変数が有する複数の状態がグループ化され、複数の状態が1つの状態として処理されることにより、動作モデルの状態数が削減される。この結果、パイプラインの構成、及びパイプライン上の命令の流れに関する仕様を維持したまま、プロセッサの簡約化された動作モデル 205 を得ることができる。そして、このような動作モデル 205 に基づいて試験命令列を生成するための処理に擁する時間を短縮することができる。

【0052】本発明の第3の態様では、入力された動作モデル 301 及びその動作モデル 301 の試験状態 302 に基づいて、試験状態 302 を含まない状態遷移により試験状態 302 に到達する試験命令列 305 を列挙する処理と、試験状態 302 の次時刻（次の命令サイクル）の状態を求める処理が繰り返し実行されることにより、試験命令列 305 が自動的に生成される。次時刻に複雑な状態遷移がある場合が試験状態 302 として設定されることにより、試験状態 302 を含まない状態遷移は単純なものとなっており、試験命令列 305 を効率良く抽出することができる。

【0053】本発明の第4の態様では、動作モデル 401、407 に対して試験項目を指定するための入力情報として、或る1つの時刻における試験状態の集合ではなく、状態評価関数 402、408 を指定できる。

【0054】パイプラインプロセッサでは、トラップ制御において、トラップ発生時にトラップを誘発させた命令が実行されるより前に実行された命令と、それより後に実行された命令を正確に区別して制御する機構が、重

要な役割を果たす。特に、厳密トラップ（precisetrap）の実現のためには、トラップを誘発させた命令が実行されるより前に実行された命令は、トラップハンドラに制御が移った時点において実行を完了していなければならない。かつ、トラップを誘発させた命令が実行されるより後に実行されるはずの命令は、トラップハンドラに制御が移った時点において実行が開始されてはならない。このような状態を検証するためには、例えば、パイプラインにできる限り多くの命令が詰った状態かつトラップを発生させる状態が試験項目となるように、試験命令列を生成する必要がある。

【0055】本発明の第4の態様では、このような定性的に表現される試験項目を、状態を評価する状態評価関数 402、408 によって効率的に定義することができ、それに対応する試験命令列 505、513 を、自動的にかつ効率的に生成することができる。

【0056】本発明の第5の態様では、動作モデル 501、507 に対して試験項目を指定するための入力情報として、状態の時系列 502、508 を指定できる。例えば或る種のパイプラインプロセッサにおいて、命令の実行結果の書込みユニットとレジスタファイルとの間にバッファが設けられ、このバッファによって実際のライトバック動作がプログラム上のステップの順番に対応するように修正されることにより、正確なトラップが実現されるように設計される場合がある。このようなパイプライン機構を検証するためには、数命令サイクルの間命令の完了順序が入れ替わる状態を発生させ、バッファにデータを保持させた後に、トラップを発生させるような試験命令列を生成する必要がある。

【0057】本発明の第5の態様では、このような状態の変化を、状態の時系列 502、508 として指定することができ、一定期間にわたり特定の状態を発生させるような試験命令列 505、513 を、自動的にかつ効率的に生成することができる。

【0058】本発明の第6の態様では、動作モデル 601、607 に対して試験項目を指定するための入力情報として、状態の時系列の評価関数 602、608 を指定できる。

【0059】例えば、プログラム上のレジスタ番号と物理レジスタ番号との間で動的に番号の付替えを行う、いわゆるレジスタリネーミング機構を有するパイプラインプロセッサが存在するが、このようなプロセッサにおいては、リネーミングのための物理レジスタの数（リネーミングスペース）が不足する場合が問題となる。この不足が生じないように設計されたプロセッサにおいて本当にこの不足が発生しないことを検証するためには、できるだけ多くのリネーミングスペースが消費されるような試験命令列が必要となる。

【0060】本発明の第6の態様では、このような一定期間の状態が定性的に表現される試験項目を、状態の時

系列を評価する評価関数 602、608 によって効率的に定義することができ、それに対応する試験命令列 605、613 を、自動的かつ効率的に生成することができる。

【0061】

【実施例】以下、図面を参照しながら本発明の実施例につき詳細に説明する。

＜第1の実施例＞図7は、本発明の第1の実施例の全体構成図である。この構成図によって実現される機能は、特に図示しないプロセッサとメモリ、及びそれらによって実現される制御プログラムを主な構成要素とするコンピュータシステムによって実現されている。

【0062】図7において、仕様情報保持部701は、検証の対象であるプロセッサのパイプライン構成及びプロセッサが実行するそれぞれの命令に関する仕様情報を保持する。

【0063】初期動作モデル構成部702は、仕様情報保持部701に保持された仕様情報から、パイプラインに関する初期動作モデルM₀を構成する。状態数最小化部703は、初期動作モデルM₀の状態の数を最小化することにより、動作モデルMを構成する。

【0064】入力情報保持部704は、上述の動作モデルMとそのモデルに対する試験状態の集合Hとからなる入力情報を保持する。試験命令列挙部705は、入力情報保持部704に保持された動作モデルMと試験状態の集合Hとから、入力情報保持部704に保持された動作モデルMを構成する複数の状態変数の状態が競合を起こす状態を含むことなく、動作モデルMの状態が所定の入力状態から入力情報保持部704に保持される試験状態の集合Hに含まれる試験状態のうちの何れかへ遷移する過程に対応した試験命令列707を列挙する。

【0065】次時刻状態計算部706は、入力情報保持部704に保持された動作モデルMを構成する複数の状態変数の状態が、試験命令列挙部705が列挙した試験命令列707に対応する試験状態になった後に次に遷移する状態である次時刻状態を計算し、その次時刻状態を新たな入力状態として試験命令列挙手段705に入力させる。

【0066】以上の構成を有する第1の実施例の動作について、図8～図15の動作フローチャート、図16及び図17のデータ構成図、並びに、図18～図28の動作説明図に従って、詳細に説明する。

【0067】まず、図8のステップ801で示されるように、図7に示される仕様情報保持部701に、仕様情報が入力される。図18に、仕様情報によって表されるパイプライン構成の例を示す。

【0068】パイプライン処理では、1つの命令によって実行される1つの処理過程が複数の時間的に独立した処理ステージに分割され、それぞれの命令のそれぞれの処理ステージはそれぞれに対応した処理ユニットによ

って実行される。そして、それぞれの命令サイクルにおいて、複数の処理ユニットによって、複数の命令に対応する複数の相互に異なる処理ステージが並列に実行される。

【0069】図18において、命令フェッチユニット（IFユニット）1801は、それぞれの命令の命令フェッチステージ（IFステージ）を実行するユニットである。このユニットは、特に図示しないメモリから命令をフェッチする。

【0070】命令デコードユニット（IDユニット）1802は、それぞれの命令の命令デコードステージ（IDステージ）を実行する。このユニットは、フェッチされた命令をデコードし、また、その命令に対応するオペランドをレジスタから読み出す。

【0071】算術論理演算ユニット（ALUユニット）1803は、それぞれの命令の算術論理演算ステージ（ALUステージ）を実行する。このユニットは、デコードされた命令が整数演算命令又は論理演算命令である場合に、その命令に対応する整数演算又は論理演算を実行する。

【0072】メモリアクセスユニット（MEMユニット）1804は、それぞれの命令のメモリアクセスステージ（MEMステージ）を実行する。このユニットは、ALUユニット1803によりアドレスが計算されたデータを、特に図示しないメモリから読み出し又はそのメモリに書き込む。

【0073】F₁ユニット1805、F₂ユニット1806、及びF₃ユニット1807とから構成される浮動小数点演算ユニット（FPUユニット）は、それぞれの命令の浮動小数点演算ステージ（FPUステージ）を実行する。このユニットは、デコードされた命令が浮動小数点演算命令である場合に、その命令に対応するオペランドに対して浮動小数点演算を実行する。

【0074】ライトバックユニット（WBユニット）1808は、それぞれの命令のライトバックステージ（WBステージ）を実行する。このユニットは、MEMユニット1804によって特に図示しないメモリから取得されたデータ又はALUユニット1803によって演算されたデータを、特に図示しないレジスタに書き込む。

【0075】上述のパイプラインの構成において、1命令サイクルの期間にフェッチされる命令は1個であり、FPUステージ以外のステージの実行は、1命令サイクルで終了する。FPUステージが消費する命令サイクル数は実行される命令によって異なる。1つの命令がF₁ユニット1805によって実行された後にF₃ユニット1807によって実行される場合には、その命令に対応するFPUステージは2命令サイクルを消費する。また、1つの命令がF₁ユニット1805によって実行された後にF₂ユニット1806が実行され更にその後

F₃ ユニット1807によって実行される場合には、その命令に対応するFPUステージは3命令サイクルを消費する。

【0076】上述した例で示されるようなパイプライン構成に関する仕様を所定の形式に従って記述した仕様情報が、仕様情報保持部701に入力される。次に、図19に、仕様情報によって表される命令の実行形態の例を示す。この仕様情報により示される仕様は以下のとおりである。即ち、命令Aは、命令サイクルの進行に同期して、その命令に対応するIFステージ、IDステージ、ALUステージ、及びWBステージが順に実行される。また、命令Bは、命令サイクルの進行に同期して、その命令に対応するIFステージ、IDステージ、ALUステージ、MEMステージ、及びWBステージが順に実行される。更に、命令Cは、命令サイクルの進行に同期して、その命令に対応するIFステージ、IDステージ、F₁ ユニット1805の実行タイミングとF₃ ユニット1807の実行タイミングとからなるFPUステージ、及びWBステージが順に実行される。加えて、命令Dは、命令サイクルの進行に同期して、その命令に対応するIFステージ、IDステージ、F₁ ユニット1805の実行タイミングとF₂ ユニット1806の実行タイミングとF₃ ユニット1807の実行タイミングとからなるFPUステージ、及びWBステージが順に実行される。

【0077】上述した例で示されるような命令の実行形態に関する仕様を所定の形式に従って記述した仕様情報が、仕様情報保持部701に入力される。次に、図8のステップ802で示されるように、図7に示される初期動作モデル構成部702は、仕様情報保持部701に保持された仕様情報に従って、初期動作モデルM₀を構成する。図18及び図19に示される仕様情報に基づいて、図20に概念的に示される初期動作モデルM₀が構成される。

【0078】仕様情報から初期動作モデルM₀を構成するためのアルゴリズムは、以下に示される。まず、パイプラインの構成例に関する仕様情報（図18参照）に基づいて、パイプラインを構成する各ユニットに対応して、図16に示される構造体として定義される状態変数が確保される。このとき、ユニット内部のパイプラインも考慮され、命令サイクル毎に命令を1個保持できるユニットにつき1個の状態変数が対応させられる。図18のパイプラインの構成例では、IFユニット1801、IDユニット1802、ALUユニット1803、MEMユニット1804、及びWBユニット1808のそれぞれに対応して1つずつの状態変数が確保されると共に、FPUユニットを構成するF₁ ユニット1805、F₂ ユニット1806、及びF₃ ユニット1807のそれぞれに対応して1つずつの状態変数が確保される。また、図20に示される初期動作モデルM₀の例では、番

号#1～#8で参照される円が状態変数を表しており、それぞれの円は、図18のパイプライン構成の例において番号1801～1808で参照される各ユニットに、1対1に対応している。

【0079】次に、上述のようにして確保された状態変数を構成する図16に示される各構成要素が決定される。図16において、状態変数の“前段に接続する状態変数のリスト”には、次時刻にその状態変数に対応するユニットが処理する状態を保持する可能性のあるユニットに対応する状態変数のリストが保持される。例えば図18及び図19の例において、ALUユニット1803に注目した場合に、そのユニットによって実行されるのは命令A及び命令Bという2つの状態であって、ALUユニット1803がこれらの状態を実行する現在時刻より1単位前の時刻には、IDユニット1802がこれらの状態を実行する。従って、ALUユニット1803に対応する状態変数の“前段に接続する状態変数のリスト”には、IDユニット1802に対応する状態変数が登録される。また、F₃ ユニット1807に注目した場合に、そのユニットによって実行されるのは命令C及び命令Dという2つの状態であって、F₃ ユニット1807がこれらの状態を実行する現在時刻より1単位前の時刻には、F₁ ユニット1805が命令Cを実行しF₂ ユニット1806が命令Dを実行する。従って、F₃ ユニット1807に対応する状態変数の“前段に接続する状態変数のリスト”には、F₁ ユニット1805に対応する状態変数とF₂ ユニット1806に対応する状態変数が登録される。

【0080】図16において、状態変数の“後段に接続する状態変数のリスト”には、その状態変数に対応するユニットが保持する状態を次時刻に処理する可能性のあるユニットに対応する状態変数のリストが保持される。例えば図18及び図19の例において、ALUユニット1803に注目した場合に、そのユニットにより実行されるのは命令A及び命令Bという2つの状態であって、次時刻にこれらの状態を実行する可能性のあるユニットは、WBユニット1808とMEMユニット1804である。従って、ALUユニット1803に対応する状態変数の“後段に接続する状態変数のリスト”には、WBユニット1808に対応する状態変数とMEMユニット1804に対応する状態変数が登録される。

【0081】それぞれの状態変数の上述した“前段に接続する状態変数のリスト”と“後段に接続する状態変数のリスト”は、図20に示される初期動作モデルM₀の例では、それぞれの円で示されるそれぞれの状態変数の間を結ぶ矢印を規定する。

【0082】図16において、状態変数の“入力の集合”には、その状態変数に対応するユニットにパイプラインの外部から入力される状態の集合が保持される。初期動作モデルM₀においては、命令フェッチユニットに

対応する状態変数の“入力集合”は、そのパイプラインにおいて実行可能な全ての命令の集合であり、その他のユニットに対応する状態変数の“入力集合”は、空集合である。より具体的には、例えば図18及び図19の例において、IFユニット1801に対応する状態変数の“入力集合”には、命令A、B、C、及びDという4つの状態が登録される。また、図20に示される初期動作モデルM₀の例では、IFユニット1801に対応する状態変数の“入力集合”によって、#1で参照される円で示されるIFユニット1801に対応する状態変数に入力される矢印と、その矢印に付加される状態（命令）が規定される。

【0083】図16において、状態変数の“状態の集合”には、その状態変数に対応するユニットが処理する状態の集合が保持される。なお、ここには、処理する状態（命令）が無いことを示す情報も保持される。初期動作モデルM₀においては、各状態変数の“状態の集合”は、各状態変数に対応するユニットに入力する可能性のある全ての命令と命令が無いことを示す情報とからなる。例えば図18及び図19の例において、ALUユニット1803に注目した場合に、そのユニットにより実行されるのは命令A及び命令Bという2つの状態である。従って、ALUユニット1803に対応する状態変数の“状態の集合”には、命令A、命令Bという2つの状態と、命令が無いことを示す情報が登録される。また、図20に示される初期動作モデルM₀の例では、上述した“状態の集合”は、それぞれの状態変数に対応する円の中の記号として表現される。ここで、記号A～Dは命令A～Dという4つの状態に対応し、記号○は処理する命令が無いことを示す情報に対応する。なお、後述する状態数最小化処理によって得られる動作モデルMにおいては、各状態変数の“状態の集合”には、グループ化によって得られる状態の集合が保持され得る。

【0084】図16において、状態変数の“遷移関数”は、その状態変数に対応するユニットの前段に接続されるユニットから出力され又はパイプラインの外部から入力される状態からその状態変数に対応するユニットが次時刻に処理する状態を得るための関数が保持される。例えば図18及び図19の例において、WBユニット1808に対応する状態変数の“遷移関数”によって、WBユニット1808の前段に接続されるALUユニット1803が命令Aを出力した場合にその命令Aを次時刻に処理する命令として取得すること、WBユニット1808の前段に接続されるMEMユニット1804が命令Bを出力した場合にその命令Bを次時刻に処理する命令として取得すること、及びWBユニット1808の前段に接続されるF₃ユニット1807が命令Cを出力した場合にその命令Cを次時刻に処理する命令として取得することが規定される。また、図20に示される初期動作モデルM₀の例では、上記“遷移関数”は、1つの状態変

数から他の1つの状態変数へ向かう、記号A、B、C、又はDの組合せによってラベル付けされた矢印によって表現されている。

【0085】図16において、状態変数の“動作テーブル”には、図17に示されるデータ構造を有する動作テーブルが保持される。この動作テーブルは、その状態変数に対応するユニットが状態1、状態2、状態3、・・・等として示されるそれぞれの状態にある場合（それぞれの命令を実行している場合）に、その状態変数に対応するユニットが動作1、動作2、動作3、・・・等として示されるレジスタの読出し又は書き込み等のパイプラインの制御に影響する動作を実行するか否か、又は当該ユニットが特定のハードウェア資源を占有するか否か等を定義する。例えば図18及び図19の例において、MEMユニット1804に対応する状態変数の“動作テーブル”には、全ての状態に対してメインメモリが占有される旨が設定される。この動作テーブルは、後述する状態数最小化の処理において使用される。

【0086】それぞれの状態変数に対応して図16に示されるデータ構造を構成する上述したそれぞれの集合又は“遷移関数”は、2分決定グラフ（BDD）を用いて効率良く表現し処理することができる。なお、図16に示されるデータ構造を構成するそれぞれの構成要素は、それぞれの構成要素に対応する機能が定義されるデータへのポインタであってもよい。

【0087】続いて、図8のステップ803で示されるように、図7に示される状態数最小化部703は、上述のようにして構成された初期動作モデルM₀の状態の数を最小化することにより、動作モデルMを構成する。

【0088】図9に、ステップ803の状態数最小化処理に対応する更に詳細な動作フローチャートを示す。まず、図9のステップ901において、図8のステップ802で構成された初期動作モデルM₀の各状態変数が順序付けされる。具体的には、初期動作モデルM₀を構成する入力段の状態変数から最終段の状態変数に向かって、それぞれの状態変数に、その接続順に対応する順序情報が付加される。図20に示される初期動作モデルM₀の例では、この順序付け処理により決定される状態変数の順序は、#1、#2、#3、#5、#4、#6、#7、#8の状態変数の順になる。

【0089】次に、図9のステップ902で、入力段の状態変数から最終段の状態変数に向かって順に、それぞれの状態変数における状態数が最小化される処理が実行される。図10に、この処理に対応する更に詳細な動作フローチャートを示す。

【0090】まず、図10のステップ1001で、入力段の状態変数において、“状態の集合”に含まれる状態のうち“動作テーブル”における値が同じ状態が、グループ化される。即ち、入力段の状態変数に対応するパイプラインユニットにおいて、そのユニットで実行される

状態（命令）のうち、占有する資源、又はレジスタの読出し若しくは書き込みの動作等が同じ状態がグループ化される。この結果、グループ化の対象となった状態が新たな1つの状態に置き換えられる。即ち、状態変数の“状態の集合”において、グループ化の対象となった状態が削除され、グループ化の結果を示す新たな1つの状態が登録される。また、状態変数の“動作テーブル”において、グループ化の対象となった状態に関する記述が削除され、グループ化の結果を示す新たな1つの状態に関する記述が追加される。この記述は、グループ化の対象となった状態に関して共通に記述されていた内容と同じ内容を有する。

【0091】図18の例に示されるように、通常、入力段の状態変数は命令フェッチユニット（図18ではIFユニット1801）に対応しており、このユニットによってフェッチされる全ての命令に対応する全ての状態の“動作テーブル”における値は同じである。このため、例えば図21に示されるように、#1で示される入力段の状態変数において、全ての命令（全ての状態）が新たな1つの状態Xに置き換えられる。そして、#1の状態変数の“状態の集合”において、グループ化の対象となった状態A、B、C、D（図20参照）が削除され、グループ化の結果を示す新たな1つの状態Xが登録される。また、状態変数の“動作テーブル”において、そのグループを構成する状態A、B、C、Dに関する記述が削除されて、グループ化の結果を示す新たな1つの状態Xに関する記述が追加される。この記述は、グループ化の対象となった状態A、B、C、Dに関して記述されていた共通の内容と同じ内容を有する。なお、記号○で示される、処理する状態（命令）が無いことを示す情報は、グループ化されずに残される。

【0092】次に、図9のステップ901で決定された順序に従って、入力段の次段の状態変数から最終段の状態変数に向かって、ステップ1002～ステップ1005の処理が繰り返し実行される。

【0093】ステップ1002では、図9のステップ901で決定された順序に従って、入力段の状態変数の最も近くに接続され、かつ処理されていない状態変数Uが選択される。

【0094】ステップ1003では、状態変数Uの“状態の集合”に含まれる状態のうち、状態変数Uの“前段に接続する状態変数のリスト”に含まれる状態変数の“状態の集合”において同じグループに属し、かつ状態変数Uの“動作テーブル”における値が同じ状態が、グループ化される。この結果、ステップ1001の場合と同様にして、グループ化の対象となった状態が新たな1つの状態に置き換えられる。即ち、状態変数Uの“状態の集合”において、グループ化の対象となった状態が削除されて、グループ化の結果を示す新たな1つの状態が登録される。また、状態変数Uの“動作テーブル”にお

いて、グループ化の対象となった状態に関する記述が削除され、グループ化の結果を示す新たな1つの状態に関する記述が追加される。そして、この記述は、グループ化の対象となった状態に関して記述されていた共通の内容と同じ内容を有する。上述のグループ化の処理により、グループ化が行われた状態変数の前段に接続される状態変数に対するグループ化の結果と同じ結果、又はその結果を更に細分化した結果が得られる。

【0095】ステップ1004では、ステップ1003によるグループ化の処理が試みられた状態変数の“状態の集合”に、その状態変数の前段に接続される状態変数の“状態の集合”に含まれる状態とは異なる状態が含まれる場合、即ち、ステップ1003によるグループ化の処理が試みられた状態変数の“状態の集合”に、そのグループ化の処理が試みられた状態変数の前段に接続される状態変数の“状態の集合”に含まれる状態を細分化した状態が含まれている場合は、ステップ1003によるグループ化の処理が試みられた状態変数に対して、パイプラインの外部から、上述の異なる状態の入力があると仮定され、その状態変数の“入力の集合”に、その外部入力に対応する状態が追加される。外部入力が追加される理由は、ステップ1003によるグループ化の処理が試みられた状態変数に、その状態変数の前段に接続される状態変数の“状態の集合”に含まれる状態が入力された場合において、“遷移関数”が新たな状態を決定できるようにするためである。なお、ステップ1004の処理は、ステップ1003によるグループ化の処理の結果、グループ化が実際に行われた状態変数と、グループ化が実際には行われなかった状態変数の両方を対象として実行される。

【0096】上述したステップ1002～1004の処理が、ステップ1005で、処理されていない状態変数がなくなると判定されるまで、即ち、最終段の状態変数が処理されるまで繰り返し実行される。

【0097】図10の動作フローチャートで示される図9のステップ902の入力段側からの状態数最小化の処理によって、例えば図20に示される初期動作モデルM₀から図21に示される動作モデルM₁が得られる。この動作モデルM₁において、前述したように、入力段の#1の状態変数において、状態A、B、C、Dがグループ化されることによって、それらの状態が新たな状態Xに置き換えられる。また、#2の状態変数において、状態A、Bがグループ化されることによってそれらの状態が新たな状態Yに置き換えられ、状態C、Dがグループ化されることによってそれらの状態が新たな状態Zに置き換えられる。この場合、#2の状態変数の“状態の集合”に新たに含まれた状態Y、Zは、#2の状態変数の前段に接続される#1の状態変数の“状態の集合”に含まれる状態Xとは異なる。このため、外部入力として、#2の状態変数の“入力の集合”に、状態Y、Zが追加される。更

に、#2の状態変数に対するグループ化の結果、#3の状態変数の“状態の集合”に含まれている状態A、Bは、#3の状態変数の前段に接続される#2の状態変数の“状態の集合”に含まれる状態Y、Zとは異なる結果となる。このために、外部入力として、#3の状態変数の“入力の集合”に、状態A、Bが追加される。これと同様にして、#2の状態変数に対するグループ化の結果、#5の状態変数の“状態の集合”に含まれている状態C、Dも、#5の状態変数の前段に接続される#2の状態変数の“状態の集合”に含まれる状態Y、Zとは異なる結果となる。このため、外部入力として、#5の状態変数の“入力の集合”に、状態C、Dが追加される。

【0098】上述した図9のステップ902の入力段側からの状態数最小化の処理により、それぞれの状態変数に関して、入力段側からその状態変数に対応するユニットまでの動作が全て同じ状態（命令）をグループ化することができる。この結果、後述する試験命令列を生成するための処理が必要とする時間を短縮することができる。

【0099】図9のステップ902の入力段側からの状態数最小化の処理の後、ステップ903で、最終段の状態変数から入力段の状態変数に向かって順に、それぞれの状態変数における状態数が最小化される処理が実行される。図11に、この処理に対応する更に詳細な動作フローチャートを示す。

【0100】まず、図11のステップ1101で、最終段の状態変数において、“状態の集合”に含まれる状態のうち“動作テーブル”における値が同じ状態が、グループ化される。即ち、最終段の状態変数に対応するパイプラインユニットにおいて、そのユニットで実行される状態（命令）のうち、占有する資源、又はレジスタの読出し若しくは書込みの動作等が同じ状態がグループ化される。この結果、グループ化の対象となった状態が新たな1つの状態に置き換えられる。即ち、状態変数の“状態の集合”において、グループ化の対象となった状態が削除され、グループ化の結果を示す新たな1つの状態が登録される。また、状態変数の“動作テーブル”において、グループ化の対象となった状態に関する記述が削除され、グループ化の結果を示す新たな1つの状態に関する記述が追加される。この記述は、グループ化の対象となった状態に関して共通に記述されていた内容と同じ内容を有する。

【0101】次に、図9のステップ901で決定された順序と逆の順序に従って、最終段の前段の状態変数から入力段の状態変数に向かって、ステップ1102～ステップ1104の処理が繰り返し実行される。

【0102】ステップ1102では、図9のステップ901で決定された順序と逆の順序に従って、最終段の状態変数の最も近くに接続され、かつ処理されていない状態変数Uが選択される。

【0103】ステップ1103では、状態変数Uの“状態の集合”に含まれる状態のうち、状態変数Uの“後段に接続する状態変数のリスト”に含まれる状態変数の“状態の集合”において同じグループに属し、かつ状態変数Uの“動作テーブル”における値が同じ状態が、グループ化される。この結果、グループ化の対象となった状態が新たな1つの状態に置き換えられる。即ち、状態変数Uの“状態の集合”において、グループ化の対象となった状態が削除され、グループ化の結果を示す新たな1つの状態が登録される。また、状態変数Uの“動作テーブル”において、グループ化の対象となった状態に関する記述が削除され、グループ化の結果を示す新たな1つの状態に関する記述が追加される。この記述は、グループ化の対象となった状態に関して記述されていた共通の内容と同じ内容を有する。上述のグループ化の処理により、グループ化が行われた状態変数の後段に接続される状態変数に対するグループ化の結果と同じ結果、又はその結果を更に細分化した結果が得られる。

【0104】上述したステップ1102と1103の処理が、ステップ1104で、処理されていない状態変数がなくなったと判定されるまで、即ち、入力段の状態変数が処理されるまで繰り返し実行される。

【0105】図11の動作フローチャートで示される図9のステップ903の最終段側からの状態数最小化の処理により、例えば図21に示される動作モデルM₁から図22に示される動作モデルM₂が得られる。この動作モデルM₂において、最終段の#8の状態変数において、状態A、B、C、Dがグループ化されることによって、それらの状態が新たな状態Xに置き換えられる。また、#7の状態変数において、状態C、Dがグループ化されることによってそれらの状態が新たな状態Zに置き換えられる。

【0106】上述した図9のステップ903の最終段側からの状態数最小化の処理により、それぞれの状態変数に関して、その状態変数に対応するユニットから最終段のユニットまでの動作が全て同じ状態（命令）をグループ化することができる。この結果、後述する試験命令列を生成するための処理が必要とする時間を短縮することができる。

【0107】図9のステップ903の最終段側からの状態数最小化の処理が終了することにより図8のステップ803の状態数最小化の処理が終了する。この結果、状態数最小化部703は、上述の最終的に得られる動作モデル（例えば図22に示される動作モデルM₂）を、動作モデルMとして出力する。

【0108】次に、図7の入力情報保持部704、試験命令列挙部705、及び次時刻状態計算部706とからなる部分は、上述した動作モデルMと、予め設定されている試験状態の集合Hとから、プロセッサのパイプラインを試験するための命令列である試験命令列707を

生成する。

【0109】これらの部分が実行する動作について、図12～図15に示される動作フローチャートと、図23～図28の動作説明図に従って、詳細に説明する。まず、図12のステップ1201に示されるように、図7に示される入力情報保持部704には、状態数最小化部703から出力される動作モデルMと予め設定されている試験状態の集合Hが入力され、それらが保持される。

【0110】動作モデルMは、前述した図16に示される状態変数の構造体の集合として入力情報保持部704に保持される。例えば、図18に示されるパイプライン構造に対応する動作モデルMは、図20、図21、又は図22に示される#1～#8の8つの状態変数に対応する8組の構造体の集合として保持される。

【0111】一方、試験状態の集合Hは、パイプラインハザードを発生させるような状態の集合である。例えば、理解の容易のため、図18に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図20に示される初期動作モデルM₀であると仮定した場合は、初期動作モデルM₀に対応する試験状態の集合Hの例として、図23に示される6つのケースの試験状態を設定することができる。図23において、ケース1に示される試験状態は、#6の状態変数の状態がD、かつ#5の状態変数の状態がCとなる状態である。この状態が発生する時刻（命令サイクル）の次時刻（次の命令サイクル）には、#7の状態変数において競合即ちハザードが発生する。言い換えれば、図18において、F₂ユニット1606が命令Dを実行し、かつF₁ユニット1605が命令Cを実行した命令サイクルの次の命令サイクルにおいて、F₃ユニット1607には命令Dと命令Cが同時に入力される。この場合、F₃ユニット1607は、命令Dと命令Cの競合を設計された仕様に従って解決することを要求される。ケース2～ケース6に示される試験状態も、ケース1の場合と同様に理解することができる。一般的には、“前段に接続する状態変数のリスト”（図16）に複数の状態変数が登録されている状態変数を対象として、試験状態を設定することができる。

【0112】図7の入力情報保持部704、試験命令列挙部705、及び次時刻状態計算部706とからなる部分によって自動的に生成される試験命令列707（図7）によって、例えば図23に示されるような試験状態が意図的に発生させられ、この結果、そのような試験状態に対してプロセッサが正常に機能するか否かを試験することができるのである。

【0113】動作モデルMが、図20に示されるような初期動作モデルM₀ではなく図22に示されるような状態数最小化の処理が施された後の動作モデルM₂であるような場合においても、図23の場合と同様な試験状態の集合Hを設定することができる。

【0114】図23に例示したようなそれぞれの試験状態は、図7の入力情報保持部704においては、動作モデルMを構成するそれぞれの状態変数に試験状態を対応させたデータとして保持される。例えば、動作モデルMが図18のパイプライン構造に対応している場合、図23に示されるケース1の試験状態は、図29に示されるデータ構造として入力情報保持部704に保持される。即ち、図29においては、パイプラインの各ユニットに対応する#1～#8の状態変数が定義され、#6の状態変数に状態Dが、#5の状態変数に状態Cが設定される。状態が設定されていない状態変数は任意の状態を取ることができる。図23に示されるケース2～ケース6の試験状態も、図29に示されるケース1の場合と同様のデータ構造として入力情報保持部704に保持される。これらの試験状態の集合が試験状態の集合Hである。

【0115】上述したようにして、入力情報保持部704に動作モデルMと試験状態の集合Hが保持された後、ステップ1202では、初期状態の集合S₀として1つの初期状態が定義され、この初期状態の集合S₀が図7の試験命令列挙部705へ入力される。この初期状態は、動作モデルMを構成する全ての状態変数の状態が処理する命令が無い状態となっている状態である。そして、例えば、動作モデルMが図18に示されるパイプライン構造に対応している場合は、試験命令列挙部705は、上述の初期状態を、例えば図30に示されるデータ構造として保持する。図30において、それぞれの状態変数に設定されている記号○は、それぞれの状態変数に対応するパイプラインユニットにおいて処理する命令が無いことを示す状態を表している。

【0116】続いて、図7の試験命令列挙部705は、ステップ1203において、初期状態の集合S₀に含まれる初期状態からの試験命令列挙の処理を実行する。この処理では、動作モデルMが対象とする全ての試験命令列のうちから、初期状態の集合S₀に含まれる例えば図30に示される初期状態からの基本遷移のみによって到達できる試験状態（この試験状態は、試験状態の集合Hに含まれる）を発生させる試験命令列が抽出される。ここで、基本遷移とは、動作モデルMを構成するそれぞれの状態変数における状態の変化が、パイプラインハザードを原因とするインターロックにより何れかの命令サイクルにおいて停滞するような遷移を含まない遷移をいう。

【0117】図13は、図12のステップ1203の処理を示す動作フローチャートである。まず、ステップ1301では、状態の集合Uが空集合φとされる。また、状態の集合Sに、開始状態として、図12のステップ1202で定義された初期状態の集合S₀に含まれる図30に示される初期状態が設定される。

【0118】続いて、ステップ1302～1305の一連の処理が繰り返し実行される。これらの一連の処理の

1回の実行は、動作モデルMに対応するパイプラインにおける1命令サイクルの時間経過に対応する。

【0119】上述の一連の処理のうちステップ1302では、動作モデルMにおいて、現在の時刻（命令サイクル）までに到達している状態の集合Uに、1時刻（1命令サイクル）前に始めて到達した状態の集合Sの内容が追加される。それぞれの状態の時刻は、初期状態に対応する時刻（命令サイクル）を基準とする。そして、この処理は、状態の集合Uと状態の集合Sの論理和を演算し、その演算結果を新たな状態の集合Uとする処理として表現される。なお、状態の集合Sの初期値は、ステップ1301で設定される初期状態の集合S₀である。この結果得られる状態の集合Uは、後述するステップ1304で使用される。

【0120】ステップ1303では、動作モデルMにおいて、1時刻（1命令サイクル）前に始めて到達した状態の集合Sに含まれる各状態から、基本遷移により到達する次時刻（次の命令サイクル）の各状態が計算される。そして、それぞれの状態のうち試験状態でない状態の集合が、次時刻の状態の集合S'として得られる。

【0121】このステップ1303において、基本遷移により到達する次時刻の状態が試験状態である場合には、その状態に至る命令列が図7の試験命令列挙部705から試験命令列707として出力される。このようにして、初期状態の集合S₀に含まれる例えば図30に示される初期状態からの基本遷移のみによって到達できる試験状態を発生させる試験命令列が抽出される。この場合、試験状態はパイプラインハザードを発生させるため、試験状態に対応する上述の状態は次時刻において基本遷移を起こすことはできない。このため、その状態は、次時刻の状態の集合S'には追加されない。

【0122】ステップ1304では、次時刻の状態の集合S'に含まれる状態のうち現在の時刻（命令サイクル）までに到達している状態の集合Uに含まれない状態が、次時刻に始めて到達した状態の集合Sとされる。この処理は、状態の集合S'と、状態の集合Uの排他論理の論理積を演算し、その演算結果を新たな状態の集合Sとする処理として表現される。

【0123】ステップ1305では、次時刻に始めて到達した状態の集合Sが空集合φであるか否かが判定される。ステップ1305の判定がNOならば、次時刻に始めて到達した状態の集合Sから次時刻における基本遷移によって到達できる試験状態を検索するために、ステップ1302～1305の一連の処理が繰り返される。

【0124】以上の図13のステップ1302～1305の一連の処理の繰返しによって、動作モデルMが対象とする全ての試験命令列のうちから、初期状態の集合S₀に含まれる例えば図30に示される初期状態からの基本遷移のみによって到達できる全ての試験状態を発生させる全ての試験命令列が抽出される。

【0125】図14に、図13のステップ1303の処理の動作フローチャートを示す。ステップ1401では、次時刻の状態の集合S'が空集合φとされる。ステップ1402では、1時刻（1命令サイクル）前に始めて到達した状態の集合Sから、その1つの要素である状態sと、その状態sに至る命令列T_sが取り出される。今、状態の集合Sが初期状態である場合には、状態sは、例えば図30に示される初期状態のデータとして、図7の試験命令列挙部705内の特には図示しないメモリに保持されている。また、状態sに至る命令列T_sは存在しない。なぜならば、図30に示される初期状態は、動作モデルMを構成する全ての状態変数の状態が処理する命令が無い状態となっている状態であるからである。

【0126】ステップ1403においては、図7の入力情報保持部704に保持されている動作モデルMを構成する全ての状態変数の“入力集合”（図16）に設定されている入力状態の集合が、入力状態集合Xとして定義される。なお、この入力状態集合Xには、入力される命令が無い状態も含まれる。例えば、理解の容易のため、図18に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図20に示される初期動作モデルM₀であると仮定した場合は、入力状態集合Xは、命令フェッチユニットに対応する状態変数の“入力集合”に格納されている実行可能な全ての命令の集合である。即ち、X = {A, B, C, D, O}である。ここで、記号Oは、入力される命令が無いことを示す状態に対応する。動作モデルMが、図20に示されるような初期動作モデルM₀ではなく図22に示されるような状態数最小化の処理が施された後の動作モデルM₂であるような場合においても、動作モデルM₂を構成する全ての状態変数の“入力集合”に設定されている入力状態の集合として、入力状態集合Xを定義することができる。

【0127】ステップ1404では、ステップ1403で定義された入力状態集合Xから、その1つの要素である入力状態xが取り出される。例えば、図18に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図20に示される初期動作モデルM₀であると仮定した場合において、入力状態集合X = {A, B, C, D}から入力状態xとして命令Aが取り出される。

【0128】ステップ1405では、ステップ1402において状態の集合Sから取り出された1つの状態sと、ステップ1404において入力状態集合Xから取り出された1つの入力状態xに基づいて、図7の入力情報保持部704に保持されている動作モデルMを構成するそれぞれの状態変数の“遷移関数”によって、それぞれの状態変数の次時刻（次の命令サイクル）の状態s'が計算される。また、状態sに至る命令列T_sに入力状態

xを付加した命令列が、次時刻の状態 s' に至る命令列 T_s として計算される。例えば、図18に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図20に示される初期動作モデル M_0 であると仮定した場合において、状態 s が図30のデータにより示される初期状態で、入力状態 x が命令Aである場合には、初期動作モデル M_0 である動作モデルMを構成するそれぞれの状態変数の“遷移関数”により計算される次時刻の状態 s' と、その状態に至る命令列 T_s は、図31(a)のデータによって示される。この例の場合、初期動作モデル M_0 である動作モデルMの状態が、そのモデルを構成する全ての状態変数において処理する命令が無い状態である初期状態から、図19のIFユニット1601に対応する#1の状態変数が命令Aを実行した状態に変化する。動作モデルMが、図20に示されるような初期動作モデル M_0 ではなく図22に示されるような状態数最小化の処理が施された後の動作モデル M_2 であるような場合においても、動作モデル M_2 を構成するそれぞれの状態変数の“遷移関数”により、次時刻の状態 s' と、その状態に至る命令列 T_s を計算することができる。

【0129】ステップ1406では、ステップ1405で計算された次時刻の状態 s' が、入力情報保持部704に保持されている試験状態の集合Hに含まれるか否かが判定される。例えば、図18に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図20に示される初期動作モデル M_0 であると仮定した場合において、次時刻の状態 s' が図31(a)のデータによって示される状態である場合には、この状態と、図29に示されるようなデータ形式で入力情報保持部704に保持されている例えば図23に示されるそれぞれの試験状態が、一致するか否かが検査される。なお、例えば図29に示されるケース1の試験状態（図23参照）の場合、状態が設定されていない状態変数は任意の状態を取ることができる。

【0130】ステップ1406において、次時刻の状態 s' が何れの試験状態とも一致しないと判定された場合には、ステップ1407で、次時刻の状態の集合 S' に、次時刻の状態 s' とその状態に至る命令列 T_s が追加される。例えば、次時刻の状態 s' を示す図31(a)のデータとその状態に至る命令列 T_s のデータが、図7の試験命令列挙部705内の特には図示しないメモリ内の次時刻の状態の集合 S' のリストに追加される。

【0131】ステップ1409では、ステップ1403で定義された入力状態集合Xから取り出されるべき入力状態 x がなくなったか否かが判定される。ステップ1409の判定がNOなら、再びステップ1404が実行される。例えば、図18に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図20に示される初期動作モデル M_0 であ

ると仮定した場合において、入力状態集合 $X = \{A, B, C, D\}$ から、入力状態 x として、既に取り出された命令A以外の命令Bが取り出される。

【0132】次に、再びステップ1405が実行される。例えば、図18に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図20に示される初期動作モデル M_0 であると仮定した場合において、状態 s が図30のデータによって示される初期状態であり、入力状態 x が命令Bである場合には、初期動作モデル M_0 である動作モデルMを構成するそれぞれの状態変数の“遷移関数”によって計算される次時刻の状態 s' と、その状態に至る命令列 T_s は、図31(b)のデータによって示される。

【0133】更に、再びステップ1406が実行され、例えば図31(b)のデータによって示される次時刻の状態 s' と、図29に示されるようなデータ形式で入力情報保持部704に保持されている例えば図23に示されるそれぞれの試験状態が、一致するか否かが検査される。

【0134】そして、ステップ1406において、再び、次時刻の状態 s' が何れの試験状態とも一致しないと判定された場合には、再びステップ1407が実行されることによって、次時刻の状態の集合 S' に、次時刻の状態 s' を示す例えば図31(b)のデータが追加される。

【0135】その後、ステップ1409では、再び、ステップ1403で定義された入力状態集合Xから取り出されるべき入力状態 x がなくなったか否かが判定される。上述したステップ1404～1409の一連の処理が繰り返された結果、ステップ1409において、ステップ1403で定義された入力状態集合Xから取り出されるべき入力状態 x がなくなったと判定された場合には、ステップ1410で、1時刻（1命令サイクル）前に始めて到達した状態の集合Sから取り出されるべき状態 s がなくなったか否かが判定される。今、状態の集合Sが初期状態である場合には、状態 s は、例えば図30のデータで示される1つの初期状態のみであるため、ステップ1410の判定はYESとなる。この結果、図13のステップ1303の処理が終了する。

【0136】図13のステップ1304では、上述のようにして計算された次時刻の状態の集合 S' に含まれる状態 s' とその状態に至る命令列 T_s のうち現在の時刻（命令サイクル）までに到達している状態の集合Uに含まれない状態とその状態に至る命令列が、次時刻に始めて到達した状態の集合Sとされる。入力状態 x が入力される命令が無いことを示す状態であったような場合、又はパイプラインにおいてインターロックが発生することによりパイプライン全体の動作が停止したような場合においては、次時刻の状態の集合 S' に含まれる状態が現在の時刻（命令サイクル）までに到達している状態の集

合Uに含まれる結果となる可能性がある。このため、ステップ1304の処理が必要となる。

【0137】その後、次時刻（次の命令サイクル）に対応して、再び図13のステップ1303に対応する図14の動作フローチャートが実行される。そして、ステップ1402において、1時刻（1命令サイクル）前に始めて到達した状態の集合Sから、その1つの要素である状態sとその状態sに至る命令列 T_s として、例えば図31(a)に示されるデータが取り出される。また、ステップ1404において、ステップ1403で定義された入力状態集合Xから、その1つの要素である入力状態xとして、例えば命令Aが取り出される。この例でステップ1405が実行されると、初期動作モデル M_0 である動作モデルMを構成するそれぞれの状態変数の“遷移関数”により計算される次時刻の状態 s' と、その状態に至る命令列 $T_{s'}$ は、図32(a)のデータによって示される。この例の場合、初期動作モデル M_0 である動作モデルMの状態が、図31(a)のデータで示されるように図19のIFユニット1601に対応する#1の状態変数が命令Aを実行した状態から、図32(a)のデータで示されるように図19のIDユニット1602に対応する#2の状態変数がIFユニット1601から出力された命令Aを実行し、かつIFユニット1601に対応する#1の状態変数が新たな命令Aを実行した状態に変化する。

【0138】この例においてステップ1406が実行されると、例えば図32(a)のデータによって示される次時刻の状態 s' と、図29に示されるようなデータ形式で入力情報保持部704に保持されている例えば図23に示されるそれぞれの試験状態が、一致するか否かが検査される。そして、ステップ1406において、次時刻の状態 s' が何れの試験状態とも一致しないと判定された場合には、ステップ1407が実行されることによって、次時刻の状態の集合 S' に、次時刻の状態 s' とその状態に至る命令列 $T_{s'}$ を示す、例えば図32(a)のデータが追加される。

【0139】その後、ステップ1409では、ステップ1403で定義された入力状態集合Xから取り出されるべき入力状態xがなくなったか否かが判定される。ステップ1409の判定がNOなら、再びステップ1404が実行される。上述の例において再びステップ1404が実行されると、入力状態集合Xから、入力状態xとして、既に取り出された命令A以外の命令Bが取り出される。この例でステップ1405が実行されると、初期動作モデル M_0 である動作モデルMを構成するそれぞれの状態変数の“遷移関数”により計算される次時刻の状態 s' と、その状態に至る命令列 $T_{s'}$ は、図32(b)のデータによって示される。

【0140】この例において再びステップ1406が実行されると、例えば図32(b)のデータによって示される次時刻の状態 s' と、図29に示されるようなデータ

形式で入力情報保持部704に保持されている例えば図23に示されるそれぞれの試験状態が、一致するか否かが検査される。そして、ステップ1406において、次時刻の状態 s' が何れの試験状態とも一致しないと判定された場合には、ステップ1407が実行されることによって、次時刻の状態の集合 S' に、次時刻の状態 s' とその状態に至る命令列 $T_{s'}$ を示す、例えば図32(b)のデータが追加される。

【0141】上述したステップ1404～1409の一連の処理が繰り返された結果、ステップ1409において、ステップ1403で定義された入力状態集合Xから取り出されるべき入力状態xがなくなったと判定された場合には、ステップ1410で、1時刻（1命令サイクル）前に始めて到達した状態の集合Sから取り出されるべき状態sがなくなったか否かが判定される。上述の例においてステップ1410が実行されると、状態の集合Sには、既に処理された図31(a)に示される状態のほかに、例えば図31(b)に示される状態なども存在する。従って、ステップ1410の判定はNOとなる。

【0142】ステップ1410の判定がNOなら、再びステップ1402～1409が実行される。この結果、状態sとして取り出される例えば図31(b)に示される状態と、入力状態集合Xに含まれる全ての入力状態xとから、例えば図33(a)及び(b)に示される次時刻の状態 s' とその状態に至る命令列 $T_{s'}$ が計算される。そして、それらの状態 s' は試験状態の集合Hには含まれないため、それらの状態 s' とその状態に至る命令列 $T_{s'}$ が次時刻の状態の集合 S' に追加される。

【0143】以上のようにして、時刻（命令サイクル）の経過に従って現在の時刻までに到達している状態の集合Uが計算されてゆく。そして、図14のステップ1405が実行された時点において、次時刻の状態 s' が例えば図34に示されるデータとして計算された場合、この状態は図29及び図23に示されるケース1の試験状態と一致する。この結果、ステップ1406の判定がYESとなり、ステップ1408が実行される。このステップ1408では、例えば図34に示されるように、次時刻の状態 s' に至る命令列 $T_{s'} = \{D, C, A, A\}$ が、図7の試験命令列列挙部705から試験命令列707として出力される。この場合、試験状態はパイプラインハザードを発生させるため、試験状態に対応する図34に示される次時刻の状態 s' は更にその次の時刻において基本遷移を起こすことはできない。このため、図34に示されるその状態とその状態に至る命令列は、次時刻の状態の集合 S' には追加されない。

【0144】ステップ1408の処理の後には、ステップ1409が実行され、次時刻における別の状態 s' が探索される。ステップ1402～1410の一連の処理が繰り返され、状態の集合Sから取り出されるべき状態sがなくなったと判定されると、図13のステップ130

3の処理が終了する。

【0145】上述の図14の動作フローチャートの処理を含む図13のステップ1302~1305の一連の処理の繰返しにより示される図12のステップ1203の処理が図7の試験命令列挙部705において実行されることにより、動作モデルMが対象とする全ての試験命令列のうちから、初期状態の集合 S_0 に含まれる、例えば図30に示される初期状態からの基本遷移のみにより到達できる全ての試験状態を発生させる全ての試験命令列707が抽出される。図24(a)~(d)に、図30に示される初期状態からの基本遷移のみによって到達できる図23に示されるケース1の試験状態を発生させる試験命令列707の例を示す。図24(a)のケース1-1において示されている試験命令列={D、C、A、A}が、図34の例に対応する。

【0146】試験状態の集合 H_i に図23に示される試験状態が含まれる場合には、図23のケース2の試験状態も図30に示される初期状態からの基本遷移のみによって到達することができる。そして、特には図示しないが、この試験状態を発生させる試験命令列707も、図7の試験命令列挙部705が図12のステップ1203の処理を実行することにより、抽出することができる。

【0147】図7の試験命令列挙部705が上述した図12のステップ1203の処理を実行することにより抽出される試験命令列707は、入力情報保持部704に入力される試験状態の集合 H_i に含まれる例えば図23として示される全ての試験状態を発生させるものではなく、初期状態からの基本遷移のみによって到達できる試験状態を発生させるものである。そこで、図7の試験命令列挙部705と次時刻状態計算部706は、更に、図12のステップ1204~1211を実行することにより、初期状態からの基本遷移のみにより到達できる試験状態以外の試験状態を発生させる試験命令列707を抽出する。

【0148】具体的には、まず、図7の次時刻状態計算部706が、図7の試験命令列挙部705が図12のステップ1203の処理を実行することにより抽出したそれぞれの試験命令列707に対応する試験状態の発生時刻（発生命令サイクル）の次時刻（次の命令サイクル）の状態を、動作モデルMを構成するそれぞれの状態変数の“遷移関数”として設定されている設計仕様に従って計算する。この場合に、動作モデルMを構成する状態変数のうち上述の試験状態に関係する状態変数においては、その状態変数の“遷移関数”として設定されている設計仕様に従って、上述の試験状態によって引き起こされるパイプラインハザードが解決されるようにその状態変数自身の状態が変化させられる。

【0149】その後、図7の試験命令列挙部705は、次時刻状態計算部706によって計算された新たな

状態からの基本遷移のみにより到達できる試験状態を発生させる試験命令列707を抽出する。

【0150】上述の動作を実現する図12のステップ1204~1211の処理の詳細について、以下に説明する。なお、ステップ1204~1208、及びステップ1210、1211は、図7の次時刻状態計算部706が実行し、ステップ1209は、図7の試験命令列挙部705が実行する。

【0151】まず、ステップ1204において、図7の試験命令列挙部705が前述した図12のステップ1203の処理を実行することによって抽出した試験命令列707に対応する試験状態が、試験状態の集合 H_i に含ませられる。そして、入力情報保持部704に保持されている試験状態の集合 H_i に含まれ、かつ、試験状態の集合 H_i に含まれない試験状態が、試験状態の集合 $\sim H_i$ に含ませられる。なお、記号“ $\sim H_i$ ”は、図12のステップ1204中では、記号“ H_i ”の上に記号“ \sim ”が付加された記号として表現されている。この試験状態の集合 $\sim H_i$ は、後述するステップ1210で使用される。

【0152】次に、ステップ1205で、繰返しの回数を示す変数 i の値が、1に初期設定される。その後、ステップ1211で変数 i の値が+1ずつインクリメントされながら、ステップ1206~1211の一連の処理が繰返し実行される。

【0153】まず、ステップ1206において試験状態の集合 H_i に試験状態が1つも含まれなくなると判定された場合、或いは、ステップ1207において試験状態の集合 $\sim H_{i+1}$ に試験状態が1つも含まれなくなると判定された場合には、試験命令列707の抽出処理が終了する。ここで、 H_i の初期値は H_1 、 $\sim H_i$ の初期値は $\sim H_1$ である。これらは、前述したステップ1204において計算される。その後、 H_i 及び $\sim H_i$ は、ステップ1210において計算される。

【0154】次に、図7の次時刻状態計算部706は、ステップ1208において、試験状態の集合 H_i の次時刻（次の命令サイクル）の状態の集合 K_i を計算する。図15に、図12のステップ1208の処理の動作フローチャートを示す。

【0155】ステップ1501では、次時刻の状態の集合 K_i が空集合 ϕ とされる。ステップ1502では、処理対象である試験状態の集合 H_i から、その1つの要素である状態 h と、その状態 h に至る命令列 T_h が取り出される。例えば、図18に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図20に示される初期動作モデル M_0 であると仮定した場合において、図12における変数 i の値が1である場合には、試験状態の集合 $H_i = H_1$ に含まれる状態 h に至る命令列 T_h の一部には、図24(a)~(d)に示されるように、前述した図12のステップ1

203において抽出された80種類の試験命令列707が含まれる。また、試験状態の集合 $H_i = H_1$ の要素の一部には、状態 h として、図24(a)～(d)に示されるように、上述の80種類の試験命令列707に対応する80種類の試験状態が含まれる。これらの試験状態は、図23のケース1の試験状態に対応する。更に、特には図示しないが、上述の例において、試験状態の集合 H_i に含まれる状態 h とその状態 h に至る命令列 T_h として、図23のケース2の試験状態に対応するものも存在する。

【0156】ステップ1503においては、前述した図14のステップ1403の場合と同様にして、図7の入力情報保持部704に保持されている動作モデル M を構成する全ての状態変数の“入力集合”（図16）に設定されている入力状態の集合が、入力状態集合 X として定義される。

【0157】ステップ1504では、前述した図14のステップ1404の場合と同様にして、ステップ1503で定義された入力状態集合 X から、その1つの要素である入力状態 x が取り出される。

【0158】ステップ1505では、ステップ1502において試験状態の集合 H_i から取り出された1つの状態 h と、ステップ1504において入力状態集合 X から取り出された1つの入力状態 x に基づいて、図7の入力情報保持部704に保持されている動作モデル M を構成するそれぞれの状態変数に設計仕様として設定されている“遷移関数”によって、それぞれの状態変数の次時刻（次の命令サイクル）の状態 k が計算される。また、状態 h に至る命令列 T_h に入力状態 x を付加した命令列が、次時刻の状態 k に至る命令列 T_k として計算される。この場合、動作モデル M を構成する状態変数のうち上述の試験状態 h に関係する状態変数においては、その状態変数の“遷移関数”として設定されている設計仕様に従って、上述の試験状態 h によって引き起こされるパイプラインハザードが解決されるようにその状態変数自身の状態が変化させられる。例えば、図18に示されるパイプライン構造に対応する動作モデル M が前述した状態数最小化の処理が施されていない図20に示される初期動作モデル M_0 であると仮定した場合において、図12における変数 i の値が1であって、かつ、状態 h が図24(a)～(d)の何れかの例に含まれる場合は、図19のF3ユニット1607に対応する#7の状態変数の“遷移関数”が、図19のF1ユニット1605に対応する#5の状態変数から#7の状態変数に命令Cが入力されると同時に図19のF2ユニット1606に対応する#6の状態変数から#7の状態変数に命令Dが入力されるというパイプラインハザードを解決するように、#7の状態変数自身の状態を変化させる。より具体的には、例えば上記2つの命令のうち先にパイプラインでの実行が開始された命令（図24の例では命令D）を先に実行するという関

数規則を有する“遷移関数”が適用される。動作モデル M が、図20に示されるような初期動作モデル M_0 でなく図22に示されるような状態数最小化の処理が施された後の動作モデル M_2 であるような場合においても、動作モデル M_2 を構成するそれぞれの状態変数の“遷移関数”により、次時刻の状態 k と、その状態に至る命令列 T_k を計算することができる。

【0159】ステップ1506では、次時刻の状態の集合 K_i に、次時刻の状態 k とその状態に至る命令列 T_k が追加される。これらのデータは、図7の次時刻状態計算部706においては、例えば前述した図31(a)のデータの形式と同様の形式で処理され、これらが次時刻状態計算部706内の特には図示しないメモリ内の次時刻の状態の集合 K_i のリストに追加される。

【0160】ステップ1507では、ステップ1503で定義された入力状態集合 X から取り出されるべき入力状態 x がなくなったか否かが判定される。ステップ1507の判定がNOなら、再びステップ1504～1506の一連の処理が実行される。

【0161】上述したステップ1504～1507の一連の処理が繰り返された結果、ステップ1507において、ステップ1503で定義された入力状態集合 X から取り出されるべき入力状態 x がなくなったと判定された場合には、ステップ1508で、処理対象である試験状態の集合 H_i から取り出されるべき状態 h がなくなったか否かが判定される。

【0162】ステップ1508の判定がNOなら、再びステップ1502～1507が実行される。この結果、状態 h として取り出される例えば図24(a)～(d)に示されるそれぞれの試験状態と、入力状態集合 X に含まれる全ての入力状態 x とから、次時刻の状態 k とその状態に至る命令列 T_k が計算される。そして、それらの状態 k が次時刻の状態の集合 K_i に追加される。

【0163】処理対象である試験状態の集合 H_i から取り出されるべき状態 h がなくなり、ステップ1508の判定がYESとなると、図12のステップ1208の処理が終了する。

【0164】ここで、例えば、図18に示されるパイプライン構造に対応する動作モデル M が前述した状態数最小化の処理が施されていない図20に示される初期動作モデル M_0 であると仮定した場合において、図12における変数 i の値が1であり、かつ、試験状態の集合 $H_i = H_1$ の要素の一部に図24(a)～(d)に示される試験状態が含まれる場合は、上記ステップ1208の処理が実行される結果、図24(a)～(d)に示される試験状態が発生した時刻の次時刻の状態 k と、その状態 k に至る命令列 T_k として、図25(a)～(d)に示されるものが計算される。なお、図24(a)～(d)に示される各ケースから図25(a)～(d)に示される各ケースへの遷移関係は、図26に示される。

【0165】次に、図7の試験命令列挙部705は、図12のステップ1209で、次時刻状態計算部706が図12のステップ1208において計算した新たな状態の集合 K_i に含まれるそれぞれの状態からの試験命令列挙の処理を実行する。この処理では、動作モデルMが対象とする全ての試験命令列のうちから、状態の集合 K_i に含まれるそれぞれの状態からの基本遷移のみによって到達できる試験状態を発生させる試験命令列707が抽出される。図12のステップ1209の処理は、図12のステップ1203の処理と全く同様の、図13及び図14の動作フローチャートにより示される。この処理の結果、例えば、図25(a)～(d)に示される状態からの基本遷移のみによって到達できる図23に示されるケース5の試験状態を発生させる試験命令列707の例として、図27(a)～(d)に示されるものが計算される。なお、図25(a)～(d)に示される各ケースから図27(a)～(d)に示される各ケースへの遷移関係は、図28に示される。

【0166】以上のようにして、図7の試験命令列挙部705から新たな試験命令列707が出力される。続いて、図7の次時刻状態計算部706は、図12のステップ1210において、図7の試験命令列挙部705が上述した図12のステップ1209の処理を実行することにより抽出した試験命令列707に対応する試験状態が、試験状態の集合 H_{i+1} に含ませられる。そして、今まで抽出されていない試験状態の集合 $\sim H_i$ に含まれ、かつ、新たに抽出された試験状態の集合 H_{i+1} に含まれない試験状態が、試験状態の集合 $\sim H_{i+1}$ に含ませられる。なお、記号“ $\sim H_{i+1}$ ”は、図12のステップ1210中では、記号“ H_{i+1} ”の上に記号“ \sim ”が付加された記号として表現されている。

【0167】その後、ステップ1211で変数 i の値が+1インクリメントされた後に、ステップ1206～1211の処理が繰り返し実行される。以上の一連の処理が、ステップ1206又は1207での判定がYESとなるまで繰り返し実行されることにより、試験命令列挙部705は、入力情報保持部704に保持されている試験状態の集合 H に含まれる全ての試験状態に対応する試験命令列707を出力することができる。

＜本発明の第2の実施例＞次に、本発明の第2の実施例について説明する。

【0168】第2の実施例では、動作モデルに対して試験項目を指定するための入力情報として、第1の実施例におけるような或る1つの時刻における試験状態の集合ではなく、状態評価関数を指定できる点が特徴である。

【0169】図35は、本発明の第2の実施例の全体構成図である。この構成は、図7に示される第1の実施例に対応する構成のうち、704～707の各部分を置き換えて得られる構成である。従って、第2の実施例において、動作モデルMを生成する部分の構成及び機能は、

特に図示しないが、第1の実施例における図7の701～703によって示される構成及び機能と同一である。

【0170】まず、入力情報保持部3501は、動作モデルMとそのモデルに対する状態評価関数とからなる入力情報を保持する。到達可能状態列挙部3502は、入力情報保持部3501に保持された動作モデルMにおいて、初期状態から到達することのできる全ての状態を列挙する。このとき同時に、各状態に対する状態評価関数も計算される。

【0171】試験命令列挙部3503は、到達可能状態列挙部3502によって列挙された、初期状態から到達可能な各状態から、それら各状態について計算される状態評価関数の値に基づいて、所定の状態を選択し、その状態に至る試験命令列3504を列挙する。

【0172】以上の構成を有する第2の実施例の動作について、図36及び図37の動作フローチャートに従って、詳細に説明する。図36は、第2の実施例の全体動作フローチャートである。

【0173】まず、図36のステップ3601に示されるように、図35に示される入力情報保持部3501には、第1の実施例における図7に示される状態数最小化部703から出力される動作モデルMと予め設定されている状態評価関数が入力されて、それらが保持される。

【0174】動作モデルMは、第1の実施例の場合と同様に、前述した図16に示される状態変数の構造体の集合として入力情報保持部3501に保持される。一方、状態評価関数は、動作モデルMの状態が入力されるとその評価値を出力する関数であり、動作モデルMによって示されるパイプライン機構の設計誤りが検出されやすい状態（パイプライン機構に負荷を与える状態）ほど大きな値を出力するように定義される関数である。

【0175】具体的には、例えばパイプラインプロセッサでは、トラップを発生させた命令より前に実行された命令はその実行を完了し、トラップを発生させた命令より後に実行されるべき命令は実行されないままであることを検証しなければならない場合がある。そのためには、パイプラインにできる限り多くの命令が詰った状態かつトラップを発生させる状態が試験項目となるように、状態評価関数が定義されればよい。そのような状態評価関数は、例えば次式によって定義される。

【0176】

$$【数1】 V(q) = N_{pipe}(q) \cdot \delta_{trap}(q)$$

ここで、 q は、動作モデルMの状態、 $N_{pipe}(q)$ は、状態 q においてパイプライン中に存在する命令の数、 $\delta_{trap}(q)$ は、トラップを発生する状態のときに値1をとり、その他の場合に値0をとるデルタ関数である。

【0177】上述の数1式で示されるような状態評価関数が、所定のデータ形式で入力情報保持部3501に保持される。上述したようにして、入力情報保持部350

1に動作モデルMと状態評価関数が保持された後、図35の到達可能状態列挙部3502によって、ステップ3602~3606の処理が実行される。これらの処理により、動作モデルMにおいて、その初期状態から到達することのできる全ての状態が列挙される。

【0178】まず、ステップ3602では、初期状態から到達することのできる状態の集合Uが空集合 ϕ とされる。また、状態の集合Sに、初期状態が設定される。この初期状態は、第1の実施例の場合と同様、動作モデルMを構成する全ての状態変数の状態が処理する命令が無い状態となっている状態である。そして、例えば、動作モデルMが図18に示されるパイプライン構造に対応している場合は、上述の初期状態は、例えば図30に示されるデータ構造として保持される。

【0179】続いて、ステップ3603~3606の一連の処理が繰り返して実行される。これらの一連の処理は、第1の実施例における図13に示される動作フローチャートのステップ1302~1305の一連の処理に対応しており、これらの一連の処理の1回の実行は、動作モデルMに対応するパイプラインにおける1命令サイクルの時間経過に対応する。

【0180】上述の一連の処理のうちステップ3603では、第1の実施例における図13のステップ1302と同様に、動作モデルMにおいて、現在の時刻（命令サイクル）までに到達している状態の集合Uに、1時刻（1命令サイクル）前に始めて到達した状態の集合Sの内容が追加される。なお、状態の集合Sの初期値は、ステップ3602で設定される初期状態の集合である。

【0181】ステップ3604では、動作モデルMにおいて、1時刻（1命令サイクル）前に始めて到達した状態の集合Sに含まれる各状態から、基本遷移により到達する次時刻（次の命令サイクル）の各状態が、次時刻の状態の集合S'として計算される。

【0182】ステップ3605においては、第1の実施例における図13のステップ1304と同様に、次時刻の状態の集合S'に含まれる状態のうち現在の時刻（命令サイクル）までに到達している状態の集合Uに含まれない状態が、次時刻に始めて到達した状態の集合Sとされる。

【0183】ステップ3606では、第1の実施例における図13のステップ1305と同様に、次時刻に始めて到達した状態の集合Sが空集合 ϕ であるか否かが判定される。

【0184】ステップ3606の判定がNOならば、次時刻に始めて到達した状態の集合Sから更に次時刻における基本遷移によって到達できる状態を検索するために、ステップ3603~3606の一連の処理が繰り返される。

【0185】以上の図36のステップ3603~3606の一連の処理の繰返しによって、例えば図30に示さ

れる初期状態からの基本遷移のみによって到達できる全ての状態が列挙される。

【0186】図37に、図36のステップ3604の処理の動作フローチャートを示す。この動作フローチャートは、第1の実施例における図14の動作フローチャートに対応している。

【0187】ステップ3701では、次時刻の状態の集合S'が空集合 ϕ とされる。ステップ3702では、1時刻（1命令サイクル）前に始めて到達した状態の集合Sから、その1つの要素である状態sと、その状態sに至る命令列T_sが取り出される。第1の実施例における図14のステップ1402と同様に、今、状態の集合Sが初期状態である場合には、状態sは、例えば図30に示される初期状態のデータとして、図35の到達可能状態列挙部3502内の特には図示しないメモリに保持されている。また、状態sに至る命令列T_sは存在しない。

【0188】ステップ3703においては、第1の実施例における図14のステップ1403と同様に、図7の入力情報保持部704に保持されている動作モデルMを構成する全ての状態変数の“入力集合”（図16）に設定されている入力状態の集合が、入力状態集合Xとして定義される。なお、この入力状態集合Xには、入力される命令が無い状態も含まれる。

【0189】ステップ3705においては、第1の実施例における図14のステップ1404と同様に、ステップ3702において状態の集合Sから取り出された1つの状態sと、ステップ3704において入力状態集合Xから取り出された1つの入力状態xに基づいて、図7の入力情報保持部704に保持されている動作モデルMを構成するそれぞれの状態変数の“遷移関数”によって、それぞれの状態変数の次時刻（次の命令サイクル）の状態s'が計算される。また、状態sに至る命令列T_sに入力状態xを付加した命令列が、次時刻の状態s'に至る命令列T_{s'}として計算される。

【0190】ステップ3706では、動作モデルMの次時刻の状態s'を入力として、状態評価関数V(s')が計算される。具体的には、状態評価関数V(s')が例えば前述の数1式で定義される場合には、まず、 $\delta \text{trap}(q)$ の値として、状態s'がトラップを発生する状態のときには1、その他の場合には0が設定される。次に、第1の実施例の場合と同様の例えば図31に示されるようなデータ形式を有する状態s'について、何れか命令に対応する記号O以外の記号が設定されている状態変数の数が計数されることにより、状態s'においてパイプライン中に存在する命令の数N_{pipe}(s')が算出される。その後、数1式に基づいて、状態評価関数V(s')が計算される。

【0191】ステップ3707では、次時刻の状態の集合S'に、次時刻の状態s'と、その状態に至る命令列

T_s と、その状態における状態評価関数 $V(s')$ の計算結果が追加される。

【0192】ステップ3708では、第1の実施例における図14のステップ1409と同様に、ステップ3703で定義された入力状態集合 X から取り出されるべき入力状態 x がなくなったか否かが判定される。

【0193】ステップ3708の判定がNOなら、再びステップ3704~3708が実行されることにより、新たな入力状態 x を入力とする、状態 s の次時刻の状態 s' と、その状態に至る命令列 T_s と、その状態における状態評価関数 $V(s')$ とが算出され、それらが次時刻の状態の集合 S' に追加される。

【0194】上述したステップ3704~3708の一連の処理が繰り返された結果、ステップ3708において、ステップ3703で定義された入力状態集合 X から取り出されるべき入力状態 x がなくなったと判定された場合には、第1の実施例における図14のステップ1410と同様に、ステップ3709で、1時刻(1命令サイクル)前に始めて到達した状態の集合 S から取り出されるべき状態 s がなくなったか否かが判定される。

【0195】ステップ3709の判定がNOなら、再びステップ3702~3708が実行される。この結果、新たな状態 s と、入力状態集合 X に含まれる全ての入力状態 x とから、状態 s の次時刻の状態 s' と、その状態に至る命令列 T_s と、その状態における状態評価関数 $V(s')$ とが算出され、それらが次時刻の状態の集合 S' に追加される。

【0196】ステップ3702~3709の一連の処理が繰り返され、状態の集合 S から取り出されるべき状態 s がなくなったと判定されると、ステップ3709の判定がYESとなって、図36ステップ3604の処理が終了する。

【0197】以上に示されるステップ3602~3606の処理が図35の到達可能状態列挙部3502により実行されることによって、動作モデル M において、その初期状態から到達することのできる全ての状態が列挙される。

【0198】その後、図36のステップ3607の処理が図35の試験命令列列挙部3503によって実行される。このステップでは、上述の一連の処理によって初期状態から到達することのできる状態の集合 U に登録されている、状態 s と、その状態に至る命令列 T_s と、その状態における状態評価関数 $V(s)$ の複数の組み合わせにおいて、状態評価関数 $V(s)$ の値が大きい順に、所定個の状態が選択され、その状態に至る命令列が上記集合 U から取り出され、それが試験命令列3504(図35)として出力される。

【0199】上述の第2の実施例により、定性的に表現されるような試験項目を状態評価関数の形式で指定することができ、それに対応する試験命令列3504を、自

動的かつ効率的に生成することができる。

＜本発明の第3の実施例＞次に、本発明の第3の実施例について説明する。

【0200】第3の実施例では、動作モデルに対して試験項目を指定するための入力情報として、状態時系列を指定できる点が特徴である。本発明の第3の実施例の全体構成図は、第2の実施例の場合と同じ図35によって示される。

【0201】第3の実施例においては、入力情報保持部3501は、動作モデル M とそのモデルに対する状態時系列の集合とからなる入力情報を保持する。到達可能状態列挙部3502は、入力情報保持部3501に保持された動作モデル M において、初期状態から到達することのできる全ての状態を列挙する。

【0202】試験命令列列挙部3503は、到達可能状態列挙部3502によって列挙された、初期状態から到達可能な各状態から、それら各状態によって構成される状態時系列のうち、入力情報保持部3501に記憶されている状態時系列の集合内の1つの状態時系列と一致するものを選択し、その状態時系列を実現する試験命令列3504を列挙する。

【0203】以上の構成を有する第3の実施例の動作について、図38~図40の動作フローチャートに従って、詳細に説明する。図38は、第3の実施例の全体動作フローチャートである。

【0204】まず、図38のステップ3801に示されるように、図35に示される入力情報保持部3501には、第1の実施例における図7に示される状態数最小化部703から出力される動作モデル M と予め設定されている状態時系列が入力され、それらが保持される。

【0205】動作モデル M は、第1、第2の実施例の場合と同様に、前述した図16に示される状態変数の構造体の集合として入力情報保持部3501に保持される。一方、状態時系列は、動作モデル M が実現する或る任意の期間の状態の時系列であり、試験項目に対応するパイプライン機構の動作条件となる一定期間の動作モデル M の状態の時系列を規定する。なお、一般に1つの試験項目に対して複数の状態時系列が存在し得る。

【0206】具体的には、或る種のパイプラインプロセッサにおいて、命令の実行結果の書込みユニット(例えば、第1の実施例に関する図18のパイプライン構成におけるライトバックユニット1808)とレジスタファイルとの間にバッファが設けられ、このバッファによって実際のライトバック動作がプログラム上のステップの順番に対応するように修正されることにより、正確なトラップが実現されるように設計される場合がある。このようなパイプライン機構を検証するためには、数命令サイクルの間命令の完了順序が入れ替わる状態を発生させ、バッファにデータを保持させた後に、トラップを発生させるような試験命令列を生成する必要がある。第3

の実施例では、このような要求を満たす状態時系列の集合が、入力情報保持部3501に入力され保持される。

【0207】上述したようにして、入力情報保持部3501に動作モデルMと状態時系列の集合が保持された後、図35の到達可能状態列挙部3502によって、ステップ3802～3806の処理が実行される。これらの処理により、動作モデルMにおいて、その初期状態から到達することのできる全ての状態が列挙される。これらの処理は、第2の実施例における図36のステップ3602～3606の一連の処理に対応している。

【0208】まず、ステップ3802では、第2の実施例における図36のステップ3602と同様に、初期状態から到達することのできる状態の集合Uが空集合 ϕ とされる。また、状態の集合Sに、初期状態が設定される。

【0209】続いて、ステップ3803～3806の一連の処理が繰り返し実行される。これらの一連の処理は、第1の実施例における図13のステップ1302～1305の一連の処理又は第2の実施例における図36のステップ3603～3606の一連の処理に対応している。

【0210】上述の一連の処理のうちステップ3803では、第1の実施例における図13のステップ1302と同様に、動作モデルMにおいて、現在の時刻（命令サイクル）までに到達している状態の集合Uに、1時刻（1命令サイクル）前に到達した状態の集合Sの内容が追加される。なお、状態の集合Sの初期値は、ステップ3802で設定される初期状態の集合である。

【0211】ステップ3804においては、動作モデルMにおいて、1時刻（1命令サイクル）前に到達した状態の集合Sに含まれる各状態から、基本遷移により到達する次時刻（次の命令サイクル）の各状態が、次時刻の状態の集合S'として計算される。

【0212】ステップ3805においては、次時刻の状態の集合S'に含まれる状態が、次時刻に到達した状態の集合Sとされる。この処理は、第1の実施例における図13のステップ1304及び第2の実施例における図36のステップ3605とは異なって、次時刻の状態の集合S'に含まれる状態が、現在の時刻（命令サイクル）までに到達している状態の集合Uに含まれていても、その状態は次時刻に到達した状態の集合Sとされる。これは、第3の実施例では、ある1時刻における状態だけではなく状態の時系列が問題となるため、状態の遷移順序を保存する必要があるためである。従って、状態の集合Uには、複数の同じ到達状態に対して異なるラベルが付与され得る。

【0213】一方において、ステップ3805では、次時刻の状態の集合S'に含まれる状態のうち現在の時刻（命令サイクル）までに到達している状態の集合Uに含まれない状態も状態S'として保存される。

【0214】そして、続くステップ3806で、この状態S'が空集合 ϕ であるか否かが判定される。即ち、どのように状態が遷移しても新しい状態が1つも発生しなくなりこのステップ3806の判定がYESとなった時点で、到達可能状態列挙部3502の動作が終了することになる。

【0215】ステップ3806の判定がNOならば、次時刻に到達した状態の集合Sから更に次時刻における基本遷移によって到達できる状態を検索するために、ステップ3803～3806の一連の処理が繰り返される。

【0216】以上の図38のステップ3803～3806の一連の処理の繰返しによって、例えば図30に示される初期状態からの基本遷移のみによって到達できる全ての状態が列挙される。

【0217】図39に、図38のステップ3804の処理の動作フローチャートを示す。この動作フローチャートは、第1の実施例における図14の動作フローチャート又は第2の実施例における図37の動作フローチャートに対応している。

【0218】ステップ3901では、次時刻の状態の集合S'が空集合 ϕ とされる。ステップ3902では、第1の実施例における図14のステップ1402と同様に、1時刻（1命令サイクル）前に到達した状態の集合Sから、その1つの要素である状態sと、その状態sに至る命令列Ts'が取り出される。

【0219】ステップ3903においては、第1の実施例における図14のステップ1403と同様に、図7の入力情報保持部704に保持されている動作モデルMを構成する全ての状態変数の“入力集合”（図16）に設定されている入力状態の集合が、入力状態集合Xとして定義される。

【0220】ステップ3905においては、第1の実施例における図14のステップ1404と同様に、ステップ3902において状態の集合Sから取り出された1つの状態sと、ステップ3904において入力状態集合Xから取り出された1つの入力状態xに基づいて、図7の入力情報保持部704に保持されている動作モデルMを構成するそれぞれの状態変数の“遷移関数”によって、それぞれの状態変数の次時刻（次の命令サイクル）の状態s'が計算される。また、状態sに至る命令列Ts'に入力状態xを付加した命令列が、次時刻の状態s'に至る命令列Ts'として計算される。

【0221】ステップ3906では、次時刻の状態の集合S'に、次時刻の状態s'と、その状態に至る命令列Ts'と、その状態の1時刻前の状態である現在時刻の状態sが追加される。現在時刻の状態sが記憶されるのは、後述する図38のステップ3807の処理において、各到達状態から順次前の状態にバックトラックできるようにするためである。

【0222】ステップ3907では、第1の実施例にお

ける図14のステップ1409と同様に、ステップ3903で定義された入力状態集合Xから取り出されるべき入力状態xがなくなったか否かが判定される。

【0223】ステップ3907の判定がNOなら、再びステップ3904~3907が実行されることにより、新たな入力状態xを入力とする、状態sの次時刻の状態s'と、その状態に至る命令列Tsと、その状態の1時刻前の状態である現在時刻の状態sとが算出され、それらが次時刻の状態の集合S'に追加される。

【0224】上述したステップ3904~3907の一連の処理が繰り返された結果、ステップ3907において、ステップ3903で定義された入力状態集合Xから取り出されるべき入力状態xがなくなったと判定された場合には、第1の実施例における図14のステップ1410と同様に、ステップ3908で、1時刻(1命令サイクル)前に到達した状態の集合Sから取り出されるべき状態sがなくなったか否かが判定される。

【0225】ステップ3908の判定がNOなら、再びステップ3902~3907が実行される。この結果、新たな状態sと、入力状態集合Xに含まれる全ての入力状態xとから、状態sの次時刻の状態s'と、その状態に至る命令列Tsと、その状態の1時刻前の状態である現在時刻の状態sとが算出され、それらが次時刻の状態の集合S'に追加される。

【0226】ステップ3902~3908の一連の処理が繰り返され、状態の集合Sから取り出されるべき状態sがなくなったと判定されると、ステップ3908の判定がYESとなって、図38ステップ3804の処理が終了する。

【0227】以上に示されるステップ3802~3806の処理が図35の到達可能状態列挙部3502により実行されることによって、動作モデルMにおいて、その初期状態から到達することのできる全ての状態が列挙される。

【0228】その後、図38のステップ3807の処理が図35の試験命令列列挙部3503によって実行される。このステップでは、上述の一連の処理によって初期状態から到達することのできる状態の集合Uに登録されている、状態sと、その状態に至る命令列Tsと、その状態の1時刻前の状態である現在時刻の状態sの複数の組み合わせを用いて、それら状態によって構成される状態時系列のうち、入力情報保持部3501に記憶されている状態時系列の集合内の1つの状態時系列と一致するものが選択されて、その状態時系列を実現する試験命令列3504が列挙される。

【0229】このステップ3807の詳細な動作フローチャートは、図40に示される。即ち、まず、ステップ4001では、図35の入力情報保持部3501に入力された状態時系列の集合のうちから1つの状態時系列が選択される。

【0230】この選択に成功しステップ4002の判定がNOとなったら、ステップ4003で、その選択された状態時系列を構成する複数の状態のうちの最終状態と同じ状態が、図38のステップ3806までの処理によって列挙された、動作モデルMにおいて初期状態から到達することのできる全ての状態の集合Uのなかから検索される。

【0231】この検索に成功しステップ4004の判定がNOとなったら、ステップ4005~ステップ4007の一連の処理の繰返しにより、その検索された状態から順次1時刻(1命令サイクル)ずつバックトラックしながら、その検索された状態を最終状態とする状態時系列が、ステップ4001によって現在選択されている状態時系列と一致するか否かが検査される。

【0232】即ち、まず、ステップ4005において、ステップ4003で検索された状態と共に集合Uに記憶されているその状態の1時刻前の状態が、ステップ4001によって現在選択されている状態時系列の最終状態の1時刻前の状態と比較される。

【0233】ステップ4006では、ステップ4005の比較結果が一致を示しているか否かが判定される。ステップ4005の比較結果が一致を示しておりステップ4006の判定がYESなら、ステップ4007で、ステップ4001によって現在選択されている状態時系列において直前に実行されたステップ4005での比較処理に使用された状態が、その状態時系列の先頭の状態であるか否かが判定される。

【0234】ステップ4007の判定がNOならば、ステップ4005~4007が再び実行されることにより、ステップ4003で検索された状態からバックトラックされた状態であって前回の比較処理に使用された状態が集合U上で検索され、その状態と共に集合Uに記憶されているその状態の更に1時刻前の状態が、ステップ4001によって現在選択されている状態時系列において前回の比較処理に使用された状態の更に1時刻前の状態と比較される。

【0235】そして、ステップ4007の判定がYESとなるまで、ステップ4005~ステップ4006の一連の処理が繰り返されることにより、ステップ4003によって検索された状態からバックトラックして得られる状態時系列が、ステップ4001によって現在選択されている状態時系列と一致するか否かが検査されるのである。

【0236】以上のようにして、最終的にステップ4007の判定がYESとなると、ステップ4003によって検索された状態からバックトラックして得られる状態時系列が、ステップ4001によって現在選択されている状態時系列と一致していることになる。

【0237】この場合には、ステップ4008で、ステップ4003によって検索された最終状態と共に集合U

に記憶されている、その最終状態に至る命令列が、図 35 の試験命令列 3504 として出力される。

【0238】その後、ステップ 4003 の処理に戻り、ステップ 4001 で選択された状態時系列を構成する複数の状態のうちの最終状態と同じ状態が、集合 U 内に更に存在するか否かが検索される。これは、図 38 のステップ 3805 の説明で前述したように、状態の集合 U には、到達経路が異なる複数の同じ到達状態に対して異なるラベルが付与され得るからである。

【0239】以上の一連の処理の繰返しにおいて、ステップ 4003 で検索された状態からバックトラックされた状態と、ステップ 4001 によって現在選択されている状態時系列上で最終状態からバックトラックされた状態とが、途中で一致しなくなると、ステップ 4006 の判定が NO となる。この場合には、ステップ 4003 で検索された状態からのバックトラックは諦められ、再びステップ 4003 の処理に戻って、ステップ 4001 で選択された状態時系列を構成する複数の状態のうちの最終状態と同じ状態が、集合 U 内に更に存在するか否かが検索される。

【0240】また、ステップ 4003 の処理の結果、ステップ 4001 で選択された状態時系列を構成する複数の状態のうちの最終状態と同じ状態が、集合 U 内にそれ以上存在しなくなると又は最初から全く存在しないと、ステップ 4004 の判定が YES となる。この場合には、ステップ 4001 の処理に戻り、図 35 の入力情報保持部 3501 に入力された状態時系列の集合のうちから別の 1 つの状態時系列が選択され、ステップ 4002 以降の処理が実行される。

【0241】最後に、ステップ 4001 の処理の結果、図 35 の入力情報保持部 3501 に入力された状態時系列の集合に、これ以上選択されるべき状態時系列が存在しなくなった場合には、ステップ 4002 の判定が YES となり、図 38 のステップ 3807 の処理を終了する。

【0242】上述の第 3 の実施例により、一定期間にわたって特定の状態を発生させるような試験命令列 3504 を、自動的かつ効率的に生成することができる。

<本発明の第 4 の実施例>次に、本発明の第 4 の実施例について説明する。

【0243】第 4 の実施例では、動作モデルに対して試験項目を指定するための入力情報として、状態時系列の評価関数を指定できる点が特徴である。本発明の第 4 の実施例の全体構成図は、第 2 の実施例の場合と同じ図 35 によって示される。

【0244】第 4 の実施例においては、入力情報保持部 3501 は、動作モデル M とそのモデルに対する状態時系列の評価関数とからなる入力情報を保持する。到達可能状態列挙部 3502 は、入力情報保持部 3501 に保持された動作モデル M において、初期状態から到達する

ことのできる全ての状態を列挙する。

【0245】試験命令列列挙部 3503 は、到達可能状態列挙部 3502 によって列挙された、初期状態から到達可能な各状態から、それら各状態によって構成される状態時系列のうち、その状態時系列に対する、入力情報保持部 3501 に記憶されている評価関数の計算結果の値が大きい順に所定個の状態時系列に対応する所定個の最終状態を選択し、各最終状態に至る試験命令列 3504 を列挙する。

【0246】以上の構成を有する第 4 の実施例の動作について、図 41 及び図 42 の動作フローチャートに従って、詳細に説明する。図 41 は、第 4 の実施例の全体動作フローチャートである。

【0247】まず、図 41 のステップ 4101 に示されるように、図 35 に示される入力情報保持部 3501 には、第 1 の実施例における図 7 に示される状態数最小化部 703 から出力される動作モデル M と予め設定されている状態時系列の評価関数が入力され、それらが保持される。

【0248】動作モデル M は、第 1 ～ 第 3 の実施例の場合と同様に、前述した図 16 に示される状態変数の構造体の集合として入力情報保持部 3501 に保持される。一方、状態時系列の評価関数は、動作モデル M が実現する或る任意の期間の状態時系列が入力されるとその評価値を出力する関数であり、動作モデル M によって示されるパイプライン機構の設計誤りが検出されやすい状態時系列（パイプライン機構に負荷を与える状態時系列）ほど大きな値を出力するように定義される関数である。

【0249】具体的には、プログラム上のレジスタ番号と物理レジスタ番号との間で動的に番号の付替えを行う、いわゆるレジスタリネーミング機構を有するパイプラインプロセッサが存在するが、このようなプロセッサにおいては、リネーミングのための物理レジスタの数（リネーミングスペース）が不足する場合は問題となる。この不足が生じないように設計されたプロセッサにおいては、本当にこの不足が発生しないことを検証するためには、できるだけ多くのリネーミングスペースが消費されるような試験命令列が必要となる。このような要求を満たす状態時系列の評価関数が、入力情報保持部 3501 に入力され保持される。より具体的には、この例の場合における評価関数は、状態時系列中に含まれる各状態において消費されるリネーミングスペースの、その状態時系列中の全状態にわたる総和を規定するような関数である。

【0250】上述したようにして、入力情報保持部 3501 に動作モデル M と状態時系列の評価関数が保持された後、図 35 の到達可能状態列挙部 3502 によって、ステップ 4102 ～ 4106 の処理が実行される。これらの処理により、動作モデル M において、その初期状態から到達することのできる全ての状態が列挙される。こ

これらの処理は、第3の実施例における図38のステップ3802~3806の一連の処理と同じである。

【0251】まず、ステップ4102では、第3の実施例における図38のステップ3802と同様に、初期状態から到達することのできる状態の集合Uが空集合 ϕ とされる。また、状態の集合Sに、初期状態が設定される。

【0252】続いて、ステップ4103~4106の一連の処理が繰り返し実行される。まず、ステップ4103では、第3の実施例における図38のステップ3803と同様に、動作モデルMにおいて、現在の時刻（命令サイクル）までに到達している状態の集合Uに、1時刻（1命令サイクル）前に到達した状態の集合Sの内容が追加される。なお、状態の集合Sの初期値は、ステップ4102で設定される初期状態の集合である。

【0253】ステップ4104においては、第3の実施例における図38のステップ3804と同様に、動作モデルMにおいて、1時刻（1命令サイクル）前に到達した状態の集合Sに含まれる各状態から、基本遷移により到達する次時刻（次の命令サイクル）の各状態が、次時刻の状態の集合S'として計算される。この処理の詳細は、第3の実施例において示した図39の動作フローチャートによって示される。

【0254】ステップ4105においては、第3の実施例における図38のステップ3805と同様に、次時刻の状態の集合S'に含まれる状態が、次時刻に到達した状態の集合Sとされる。この場合、第3の実施例の場合と同様に、次時刻の状態の集合S'に含まれる状態が、現在の時刻までに到達している状態の集合Uに含まれていても、その状態は次時刻に到達した状態の集合Sとされる。これは、第4の実施例においても、ある1時刻における状態だけではなく状態の時系列が問題となるため、状態の遷移順序を保存する必要があるためである。従って、状態の集合Uには、複数の同じ到達状態に対して異なるラベルが付与され得る。

【0255】また、ステップ4105では、第3の実施例における図38のステップ3805と同様に、次時刻の状態の集合S'に含まれる状態のうち現在の時刻までに到達している状態の集合Uに含まれない状態も状態S''として保存される。

【0256】そして、続くステップ4106で、第3の実施例における図38のステップ3806と同様に、この状態S''が空集合 ϕ であるか否かが判定される。即ち、どのように状態が遷移しても新しい状態が1つも発生しなくなりこのステップ4106の判定がYESとなった時点で、到達可能状態列挙部3502の動作が終了することになる。

【0257】ステップ4106の判定がNOならば、次時刻に到達した状態の集合Sから更に次時刻における基本遷移によって到達できる状態を検索するために、ステ

ップ4103~4106の一連の処理が繰り返される。

【0258】以上の図41のステップ4103~4106の一連の処理の繰返しによって、例えば図30に示される初期状態からの基本遷移のみによって到達できる全ての状態が列挙される。

【0259】その後、図41のステップ4107と4108の処理が図35の試験命令列挙部3503によって実行される。まずステップ4107では、上述の一連の処理によって初期状態から到達することのできる状態の集合Uに登録されている、状態sと、その状態に至る命令列Tsと、その状態の1時刻前の状態である現在時刻の状態sの複数の組み合わせを用いて、それら状態によって構成される全ての状態時系列について、その状態時系列の評価関数が計算される。

【0260】このステップ4107の詳細な動作フローチャートは、図42に示される。即ち、まず、ステップ4201では、図41のステップ4106までの処理によって列挙された、動作モデルMにおいて初期状態から到達することのできる全ての状態の集合Uのなかから、1つの到達状態が選択される。

【0261】この検索に成功しステップ4202の判定がNOとなったら、ステップ4203で、ステップ4201で選択された状態を用いて、現在バックトラックされている状態時系列の評価関数が計算される。具体的には、その状態におけるリネーミングスペースが計算され、現在バックトラックされている状態時系列の評価関数の計算結果に累算される。上述のリネーミングスペースの計算は、第1の実施例の場合と同様の例えば図31に示されるようなデータ形式を有する1つの状態について、各状態変数に対応する例えば図16及び図17に示されるデータ形式を有する動作テーブルに設定されているリネーミングスペースを、当該状態内の全状態変数にわたって合計する計算として実現できる。

【0262】その後、ステップ4204~4206の一連の処理の繰返しにより、ステップ4201で選択された状態から順次1時刻（1命令サイクル）ずつバックトラックしながら、初期状態に到達するまで、バックトラックによって生成される状態時系列に対する評価関数が順次計算される。

【0263】即ち、まず、ステップ4204において、ステップ4201で選択された状態と共に集合Uに記憶されているその状態の1時刻前の状態が選択される。次に、ステップ4205において、ステップ4203の場合と同様に、ステップ4204で選択された状態を用いて、現在バックトラックされている状態時系列の評価関数が計算される。即ち、ステップ4204で選択された状態におけるリネーミングスペースが計算され、現在バックトラックされている状態時系列の評価関数の計算結果に累算される。

【0264】ステップ4206では、ステップ4204

で選択された状態が動作モデルMにおける初期状態であるか否かが判定される。ステップ4206の判定がNOなら、ステップ4204の処理に戻り、状態時系列のバックトラックが進められる。

【0265】ステップ4206の判定がYESとなると、ステップ4201で選択された状態を最終状態とする状態時系列のバックトラックが終了し、その状態時系列に対する最終的な評価関数の計算結果が算出されたことになる。

【0266】この後、ステップ4207で、集合U内の、ステップ4201で選択された状態に対応する要素位置に、その状態を最終状態とする状態時系列に対する上述の評価関数の計算結果が登録される。

【0267】その後、ステップ4201の処理に戻り、集合Uのなかから別の1つの到達状態が選択され、ステップ4202以降の処理が繰り返される。そして、集合Uにおいてこれ以上選択されるべき到達状態が存在しなくなり、ステップ4202の判定がYESとなると、図41のステップ4107の処理を終了する。

【0268】最後に、図41のステップ4108では、上述の一連の処理によって初期状態から到達することのできる状態の集合Uに登録されている、各到達状態（各状態時系列の最終状態）と、その到達状態に至る命令列と、その到達状態を最終状態とする状態時系列の評価関数の計算結果の組合せにおいて、評価関数の値が大きい順に、所定個の到達状態が選択され、その到達状態に至る命令列が上記集合Uから取り出され、それが試験命令列3504（図35）として出力される。

【0269】上述の第4の実施例により、一定期間にわたって定性的に表現される試験状態となる試験命令列3504を、自動的かつ効率的に生成することができる。＜その他の実施態様＞最後に、本発明のその他の実施態様について説明する。

【0270】まず、本発明では、自動生成された試験命令列に、プロセッサの初期状態を設定する命令列及び終了状態を保存する命令列が付加されることにより、パイプライン制御機構の検証用の試験命令列を生成することもできる。

【0271】また、本発明は、試験命令列の自動生成のみならず、プロセッサを構成するパイプラインに対応する動作モデルに対して何等かの解析を行う一般的な技術に適用できる。特に、動作モデルに対して状態数最小化処理を実行する技術を用いることにより、その処理の結果として得られる動作モデルに対応して、最適化されたパイプラインを再構成することも可能となる。

【0272】

【発明の効果】本発明の第1～第3の態様によれば、パイプラインに関する基本的な情報を入力するだけで、自動的にプロセッサの動作モデルや試験命令列（テストプログラム）を生成することが可能となる。この結果、テ

ストプログラムの開発者は、初期設計及び設計変更に伴う工数を大幅に削減できるため、プロセッサの開発サイクル短縮に寄与するところが大きい。

【0273】特に、本発明の第2の態様により構成される動作モデルは、試験命令列の自動生成の効率化を実現するのみならず、構成される動作モデルに対応する最適化されたパイプラインを再構成することも可能となる。この結果、本発明は、初期的なハードウェア設計のための技術としても利用することができる。

【0274】また、本発明の第3の態様によれば、試験命令列302の抽出の処理を大幅に効率化することが可能となる。本発明の第4の態様によれば、トラップ制御などの定性的に表現される試験項目を、状態を評価する状態評価関数によって効率的に定義することができ、それに対応する試験命令列を、自動的かつ効率的に生成することが可能となる。

【0275】本発明の第5の態様によれば、一定期間命令の完了順序が入れ替わるような状態の時系列を状態の時系列として指定することができ、一定期間にわたって特定の状態を発生させるような試験命令列を、自動的かつ効率的に生成することが可能となる。

【0276】本発明の第6の態様では、一定期間の状態が定性的に表現される試験項目を、状態の時系列を評価する評価関数によって効率的に定義することができ、それに対応する試験命令列を、自動的かつ効率的に生成することが可能となる。

【0277】更に、本発明の全般について挙げることができる効果として、本発明により自動生成された試験命令列の集合は、それぞれ異なるパイプラインハザード発生ケースに対応させることができるのみならず、それぞれの集合から代表的な要素が取り出されることによって、様々なパイプラインハザードの発生ケースを網羅的に検証することが可能となる。

【0278】また、現在のプロセッサにおいては、命令の種類によりパイプライン経路が変更されたりレジスタ値がフォワーディング等されるため、パイプラインがインターロックを起こさない場合においても制御機構の検証が重要となる場合がある。本発明により自動生成される試験命令列の集合は、それぞれが特定のパイプライン制御パターンに対応しているため、それら制御パターンに対応して特定の制御機構が動作する状況を列挙するような作業にも役立てることができる。

【図面の簡単な説明】

【図1】本発明のブロック図（その1）である。

【図2】本発明のブロック図（その2）である。

【図3】本発明のブロック図（その3）である。

【図4】本発明のブロック図（その4）である。

【図5】本発明のブロック図（その5）である。

【図6】本発明のブロック図（その6）である。

【図7】本発明の第1の実施例の全体構成図である。

【図 8】第 1 の実施例の全体動作フローチャート（その 1）である。

【図 9】状態数最小化処理の動作フローチャートである。

【図 10】入力段側からの状態数最小化処理の動作フローチャートである。

【図 11】最終段側からの状態数最小化処理の動作フローチャートである。

【図 12】第 1 の実施例の全体動作フローチャート（その 2）である。

【図 13】試験命令列列挙処理の動作フローチャートである。

【図 14】基本遷移による次時刻の状態集合 S' の計算と試験命令列出力の処理の動作フローチャートである。

【図 15】次時刻状態集合の計算処理の動作フローチャートである。

【図 16】状態変数の構造体を示した図である。

【図 17】動作テーブルのデータ構造を示した図である。

【図 18】仕様情報によって表されるパイプライン構成の例を示した図である。

【図 19】仕様情報によって表される命令の実行形態の例を示した図である。

【図 20】初期動作モデル M_0 の例を示した図である。

【図 21】動作モデル M_1 の例を示した図である。

【図 22】動作モデル M_2 の例を示した図である。

【図 23】試験状態の集合 H の例を示した図である。

【図 24】初期状態からの基本遷移のみによって到達できるケース 1 の試験状態を発生させる試験命令列の例を示した図である。

【図 25】ケース 1-1、1-2、1-3、1-4 の状態の次時刻の状態を示した図である。

【図 26】ケース 1 とケース 1' の関係を示した図である。

【図 27】ケース 1'-1、1'-2、1'-3、1'-4 の状態からの基本遷移のみによえ到達できるケース 5 の試験状態を発生させる試験命令列の例を示した図である。

【図 28】ケース 1' とケース 5 の関係を示した図である。

【図 29】試験状態のデータ構造の例を示した図である。

【図 30】初期状態のデータ構造の例を示した図である。

【図 31】基本遷移の後の s' と T_s のデータ構造の例（その 1）を示した図である。

【図 32】基本遷移の後の s' と T_s のデータ構造の例（その 2）を示した図である。

【図 33】基本遷移の後の s' と T_s のデータ構造の例（その 3）を示した図である。

【図 34】試験状態に対応する s' と T_s のデータ構造の例を示した図である。

【図 35】本発明の第 2～第 4 の実施例の全体構成図である。

【図 36】第 2 の実施例の全体動作フローチャートである。

【図 37】第 2 の実施例における基本遷移による次時刻の状態集合 S' の計算の処理の動作フローチャートである。

【図 38】第 3 の実施例の動作フローチャートである。

【図 39】第 3 及び第 4 の実施例における基本遷移による次時刻の状態集合 S' の計算の処理の動作フローチャートである。

【図 40】第 3 の実施例における実現可能な状態時系列の計算と試験命令列出力の処理の動作フローチャートである。

【図 41】第 4 の実施例の動作フローチャートである。

【図 42】第 4 の実施例における状態時系列の選択とその評価関数の計算の処理の動作フローチャートである。

【符号の説明】

101、201	仕様情報
102、202	仕様情報保持手段
103、205、301	動作モデル
104	動作モデル構成手段
105、302	試験状態
106	試験状態列挙手段
107、305	試験命令列
108	試験命令列生成手段
203	初期動作モデル
204	初期動作モデル構成手段
206	状態数最小化手段
303	入力情報保持手段
304	入力状態
306	試験命令列列挙手段
307	次時刻状態
308	次時刻状態計算手段
401、407、501、507、601、607	動作モデル
402、408	状態評価関数
403、503、603	入力情報保持手段 403
404	到達可能状態列挙手段
405、413、505、513、605、613	試験命令列
406、506、606	試験命令列列挙手段
409、509、609	入力情報保持手段
410、510、610	到達可能状態列挙手段
411	状態評価関数計算手段
412、511、611	命令列列挙手段
414、514、614	試験命令列出力手段
502、508	状態の時系列

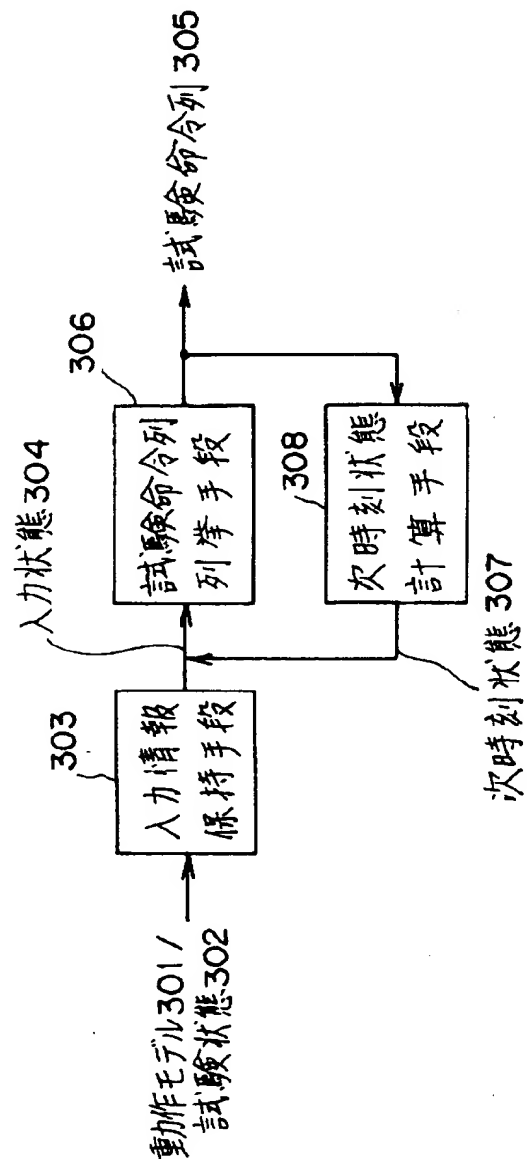
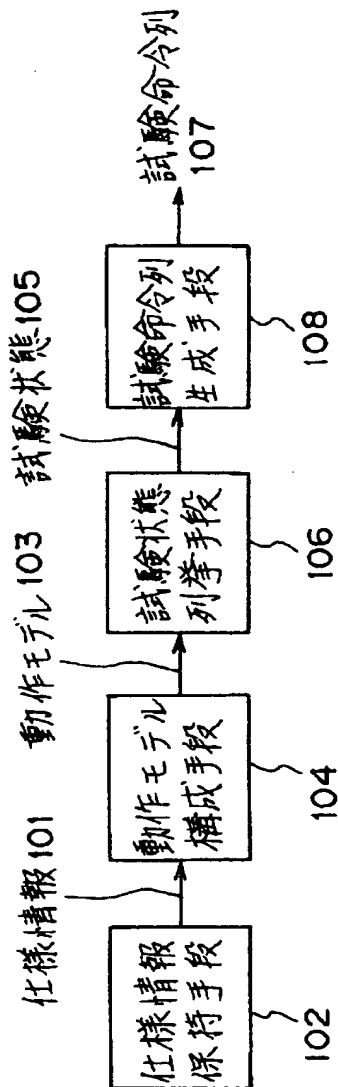
602、608 状態の時系列の評価関数
 504、604 到達可能状態時系列列挙手段
 701 仕様情報保持部
 702 初期動作モデル構成部
 703 状態数最小化部
 704、3501 入力情報保持部
 705 試験命令列列挙部

706 次時刻状態計算部
 707、3504 試験命令列
 3502 到達可能状態列挙部
 3503 試験命令列列挙部
 M₀ 初期動作モデル
 M₁、M₂、M 動作モデル
 H 試験状態の集合

【図1】

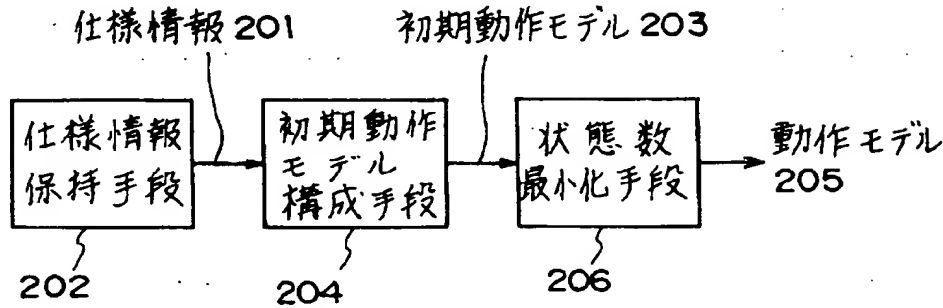
【図3】

本発明のブロック図(その1) 本発明のブロック図(その3)



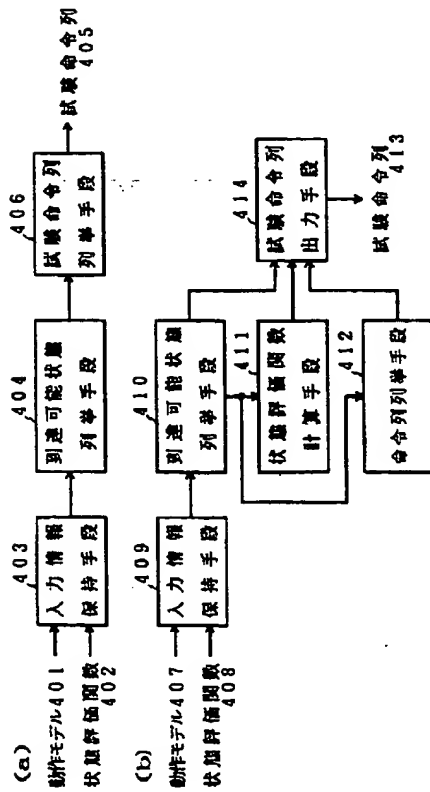
【図2】

本発明のブロック図 (その2)



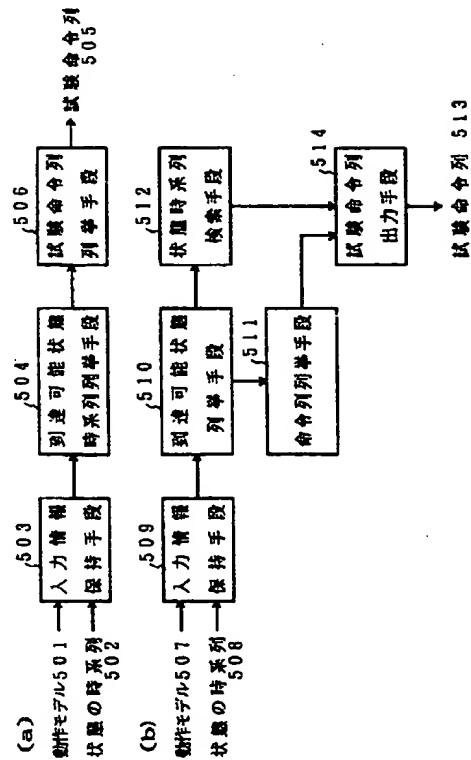
【図4】

本発明のブロック図 (その4)



【図5】

本発明のブロック図 (その5)

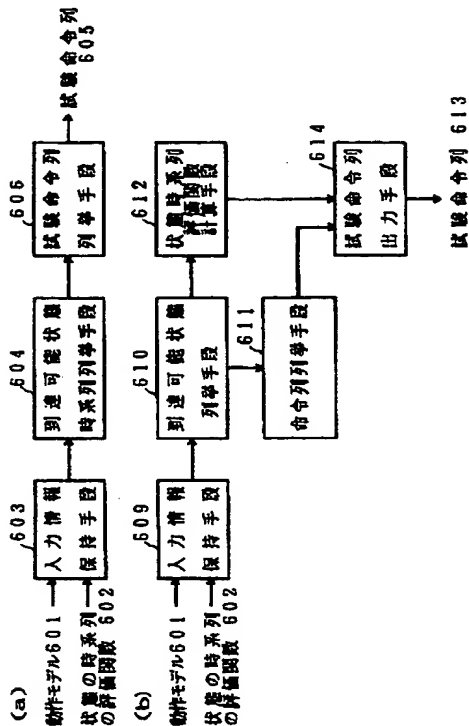


【図6】

【図8】

本発明のブロック図(その6)

第1の実施例の全体動作フローチャート(その1)



【図16】

【図17】

状態変数の構造体を示した図

動作テーブルのデータ構造を示した図

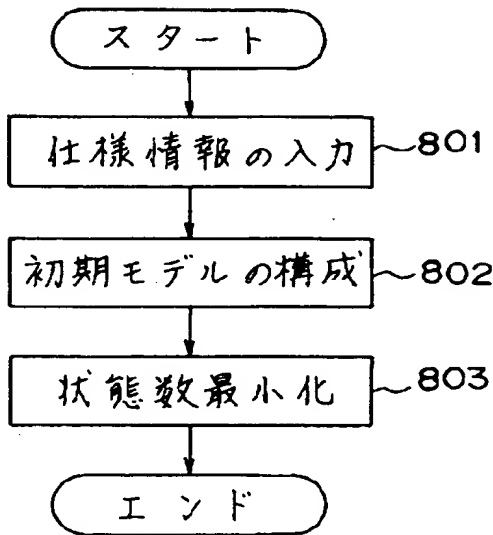
前段に接続する状態変数のリスト
後段に接続する状態変数のリスト
入力の集合
状態の集合
遷移関数
動作テーブル

	動作1	動作2	動作3	...
状態1	✓	✓		
状態2	✓	✓		
状態3	✓		✓	
⋮				

【図19】

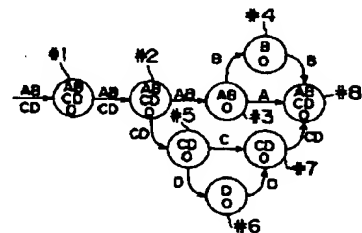
仕様情報によって表される命令の実行形態の例を示した図

命令 A: IF-ID-ALU-WB
 命令 B: IF-ID-ALU-MEM-WB
 命令 C: IF-ID-F1-F3-WB
 命令 D: IF-ID-F1-F2-F3-WB



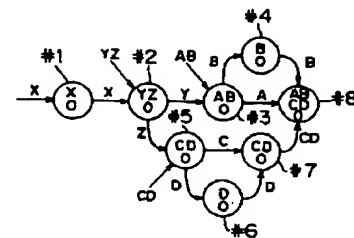
【図20】

初期動作モデルMoの例を示した図



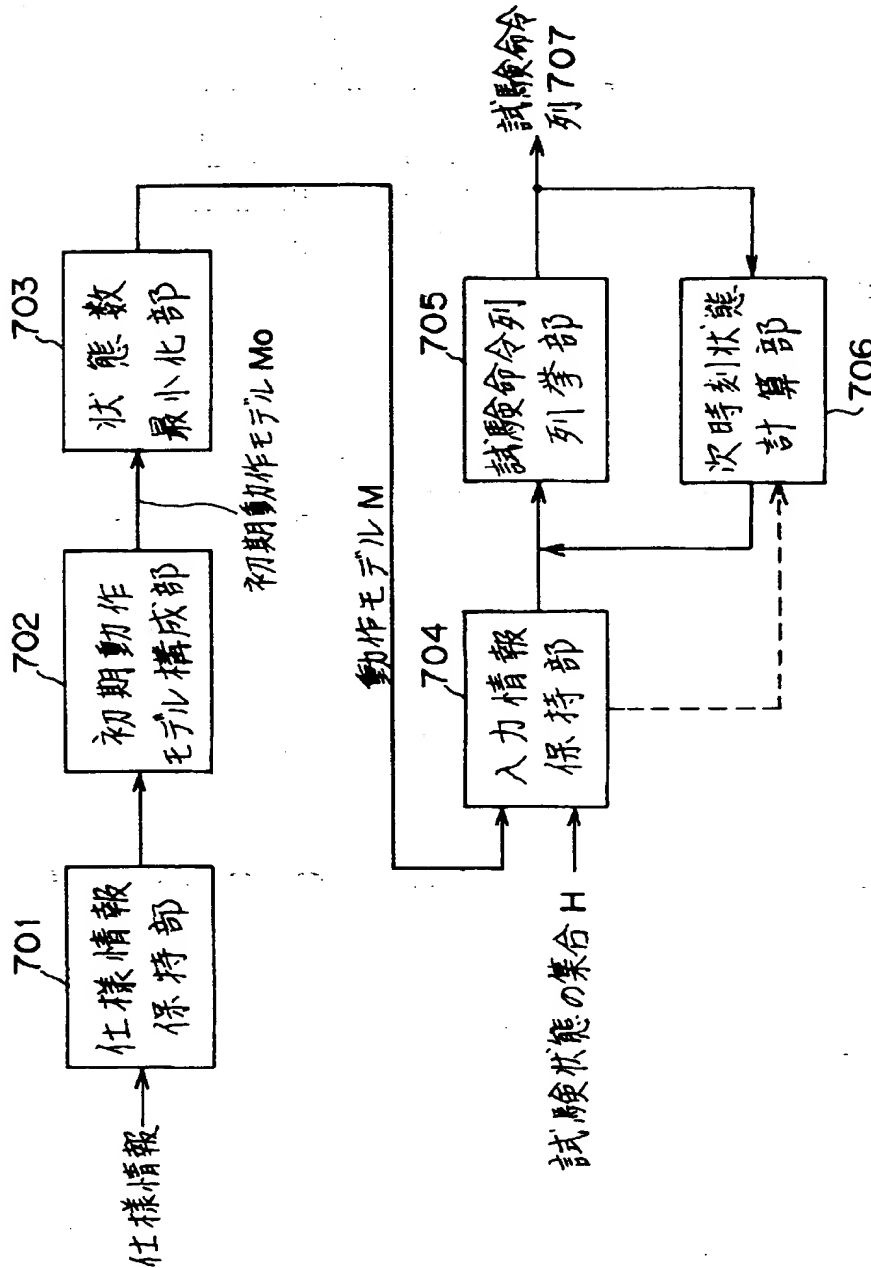
【図21】

動作モデルMiを示した図



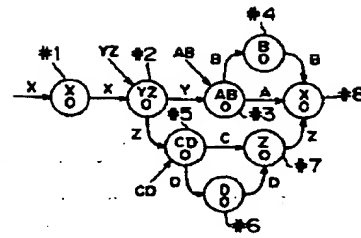
【図7】

本発明の第1の実施例の全体構成図



【図22】

動作モデル M2 を示した図



【図29】

試験状態のデータ構造の例を示した図

ケース1

状態変数	状態
#1	
#2	
#3	
#4	
#5	C
#6	D
#7	
#8	

【図30】

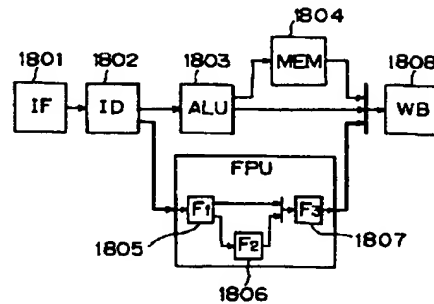
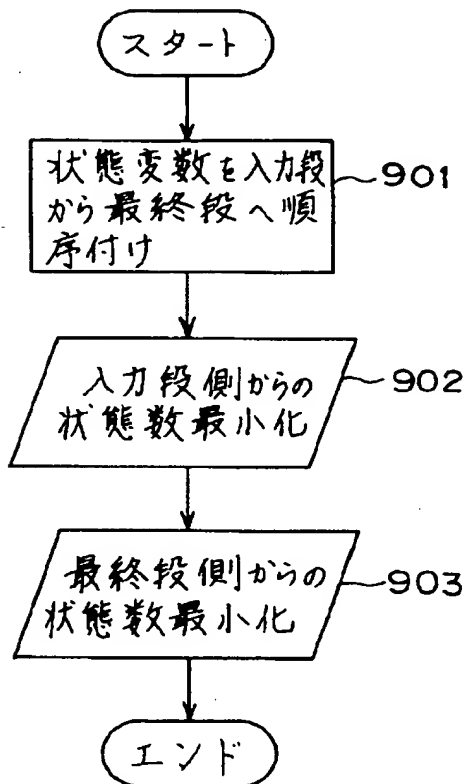
初期状態のデータ構造の例を示した図

状態変数	状態
#1	O
#2	O
#3	O
#4	O
#5	O
#6	O
#7	O
#8	O

【図 9】

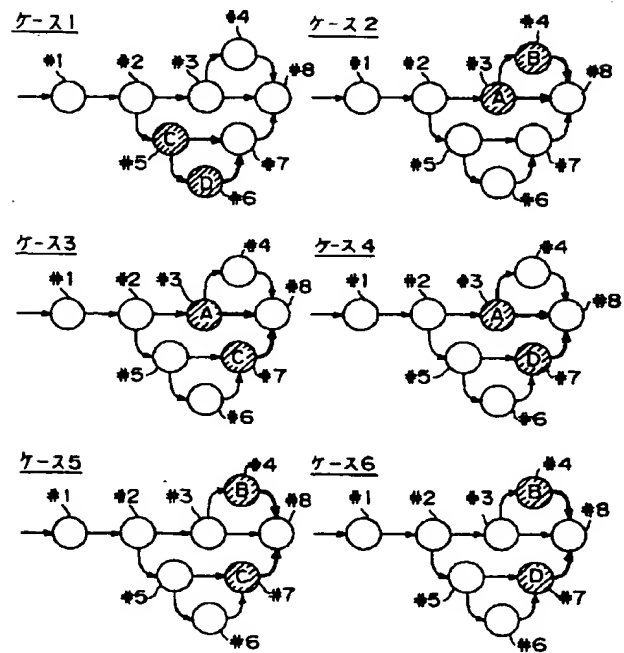
【図 18】

状態数最小化処理の動作フローチャート

仕様情報によって表される
パイプライン構成の例を示した図

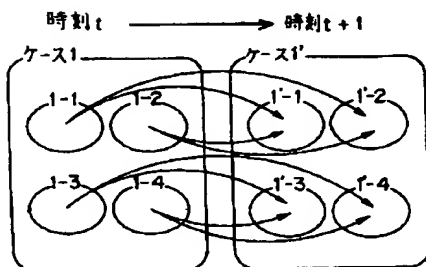
【図 23】

試験状態の集合Hの例を示した図



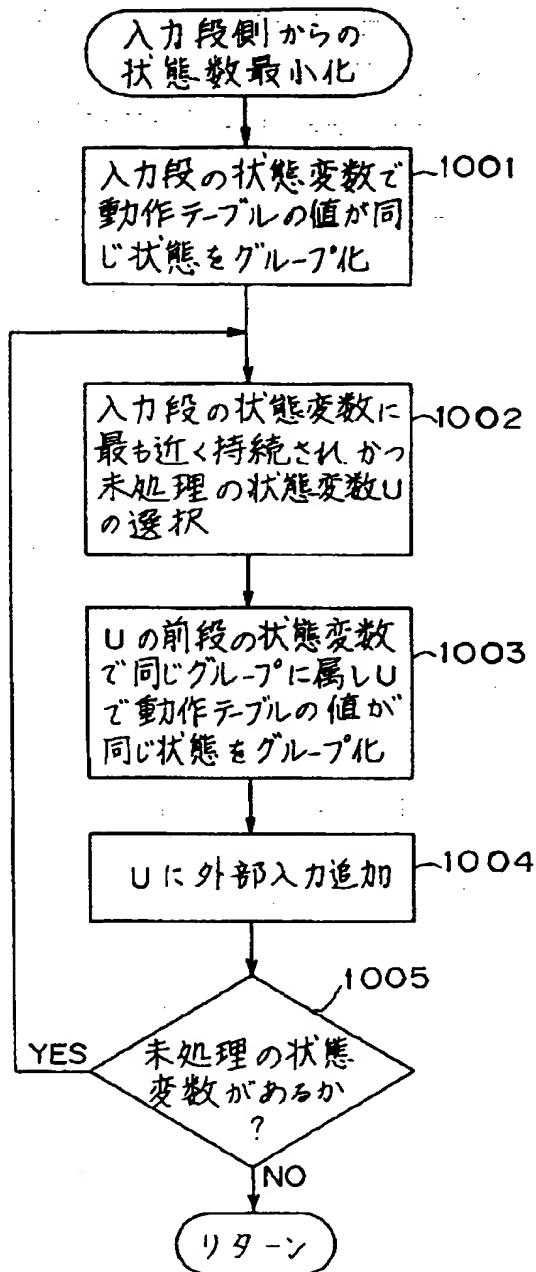
【図 26】

ケース 1 と ケース 1' の関係を示した図



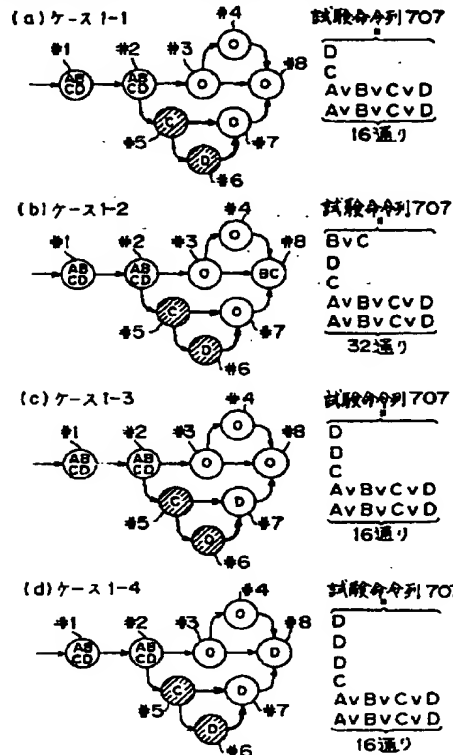
【図10】

入力段側からの状態数最小化処理の 動作フローチャート



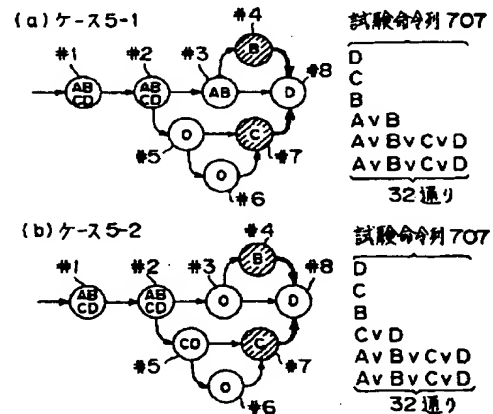
【図24】

初期状態からの基本遷移のみによって到達できるケース1の試験状態と発生させる試験命令列の例を示した図



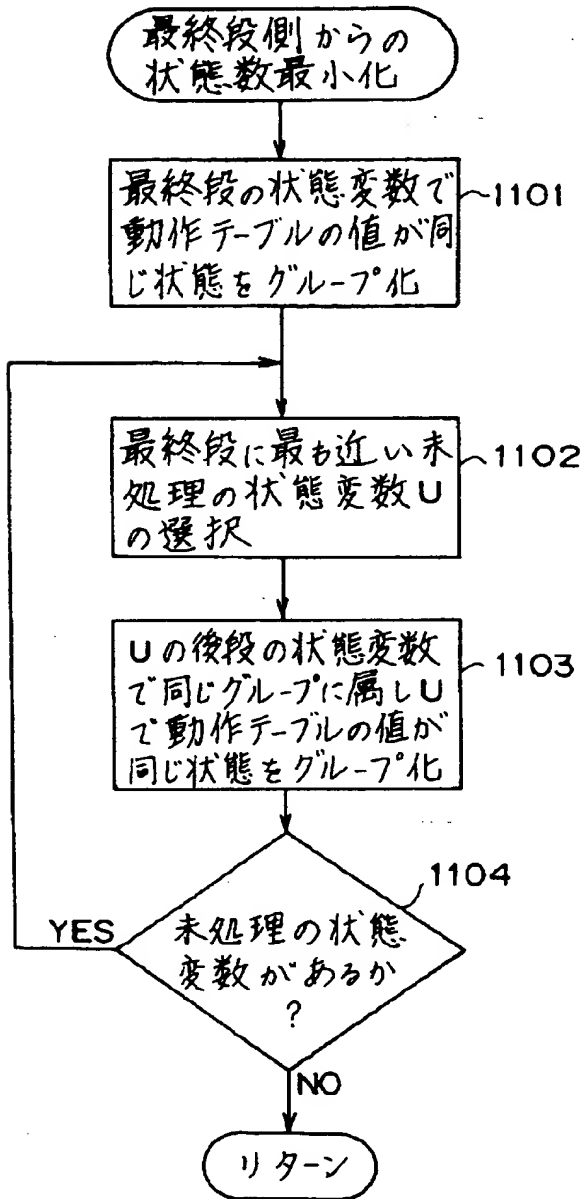
【図27】

ケース1'-1, 1'-2, 1'-3, 1'-4の状態からの基本遷移のみによって到達できるケース5の試験状態と発生させる試験命令列の例を示した図



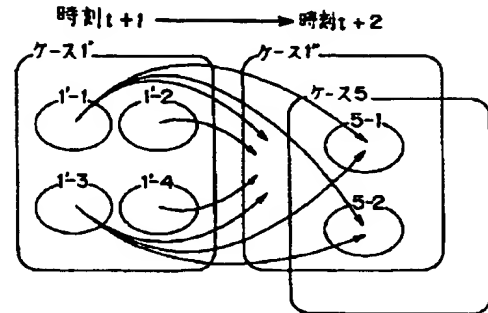
【図 11】

最終段側からの状態数最小化処理の 動作フローチャート



【図 28】

ケース1'とケース5の関係を示した図



【図 31】

基本遷移の後の S' と T_s' の
データ構造の例 (その1) を示した図

(a)	状態変数	状態	$T_s' = \{A\}$
	#1	A	
	#2	O	
	#3	O	
	#4	O	
	#5	O	
	#6	O	
	#7	O	
	#8	O	

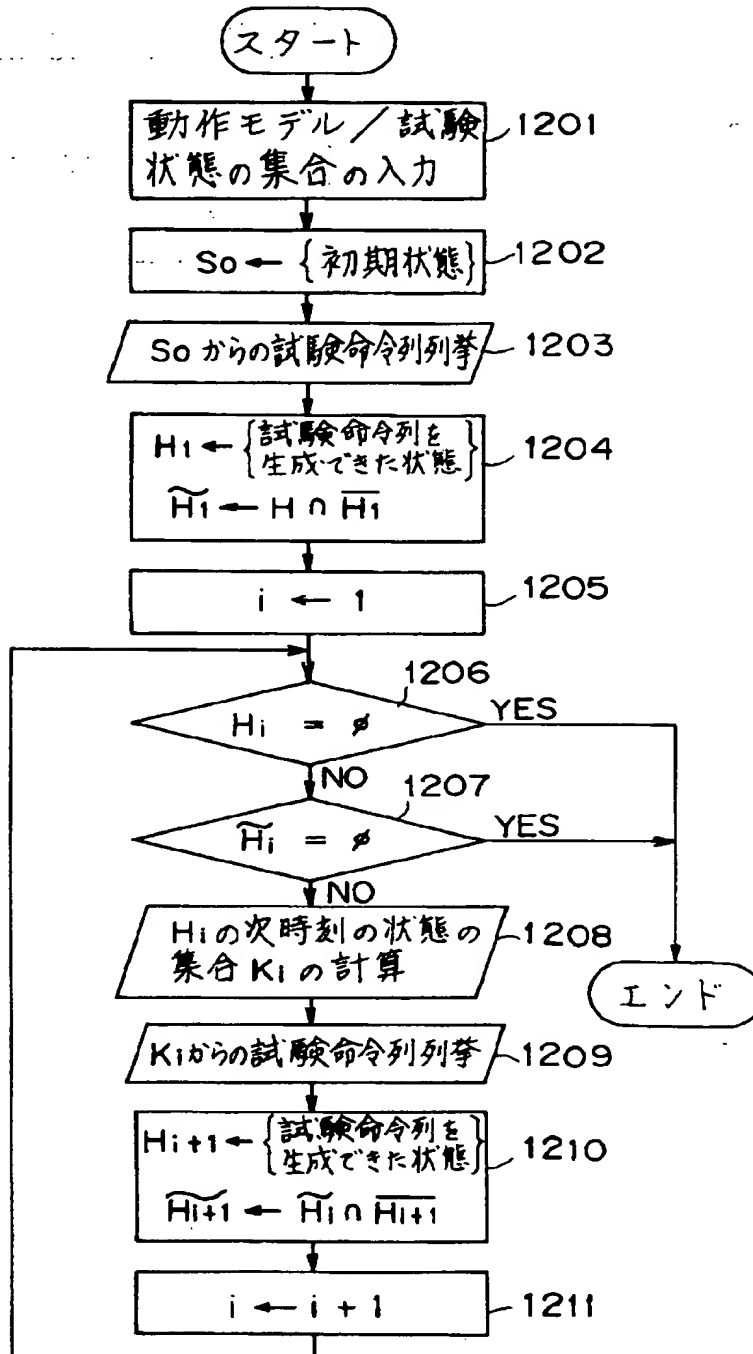
$S' :$

(b)	状態変数	状態	$T_s' = \{B\}$
	#1	B	
	#2	O	
	#3	O	
	#4	O	
	#5	O	
	#6	O	
	#7	O	
	#8	O	

$S' :$

【図12】

第1の実施例の全体動作フローチャート (その2)



【図34】

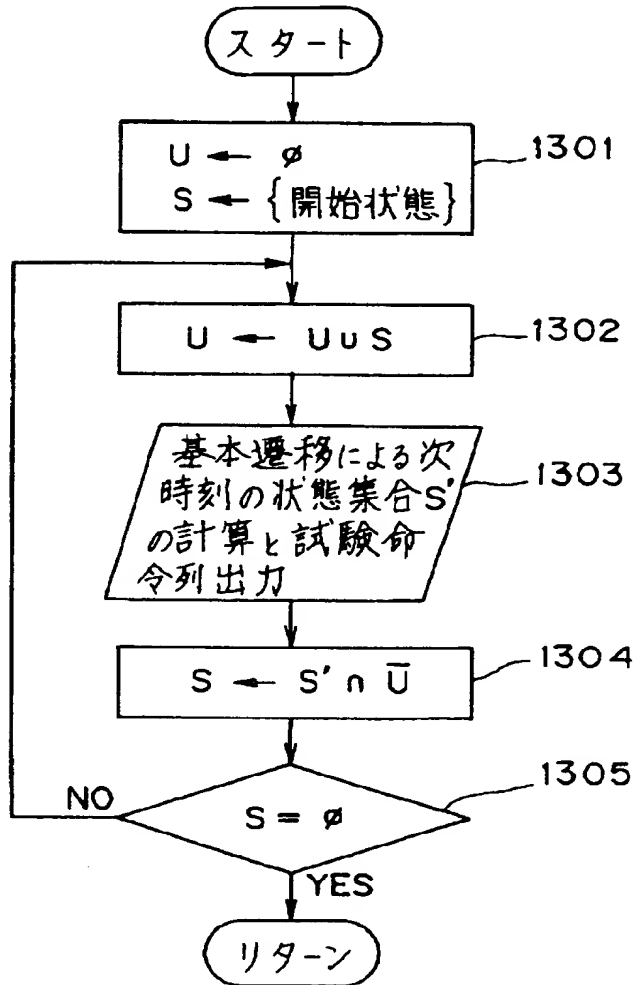
試験状態に対応する S' と T' のデータ構造の例を示した図

状態変数	状態
#1	Λ
#2	A
#3	O
#4	O
#5	C
#6	D
#7	O
#8	O

 $T' = \{D, C, A, A\}$

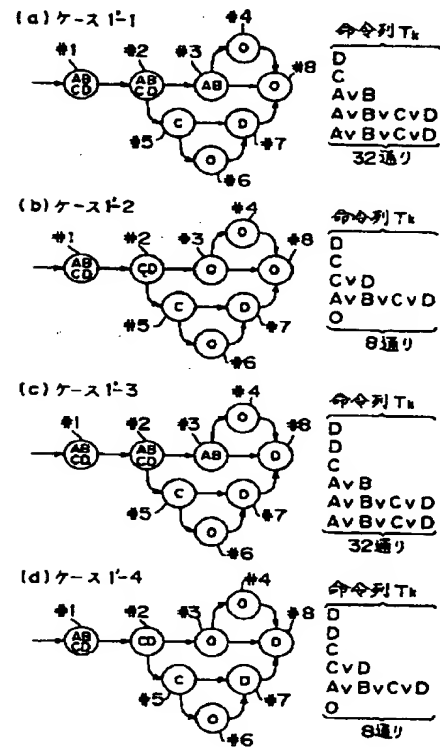
【図13】

試験命令列列挙処理の動作フローチャート



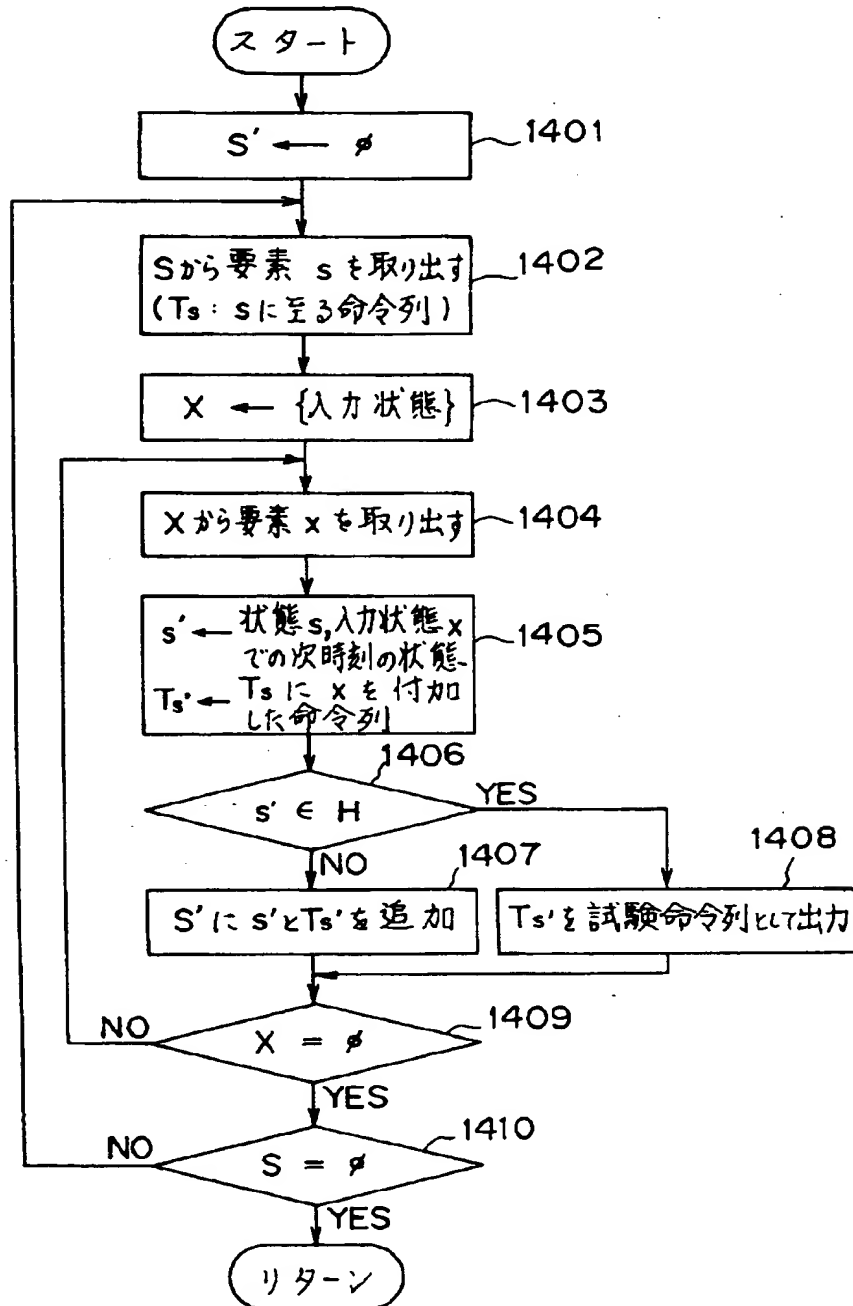
【図25】

ケース1-1, 1-2, 1-3, 1-4の状態の次時刻の状態を示した図



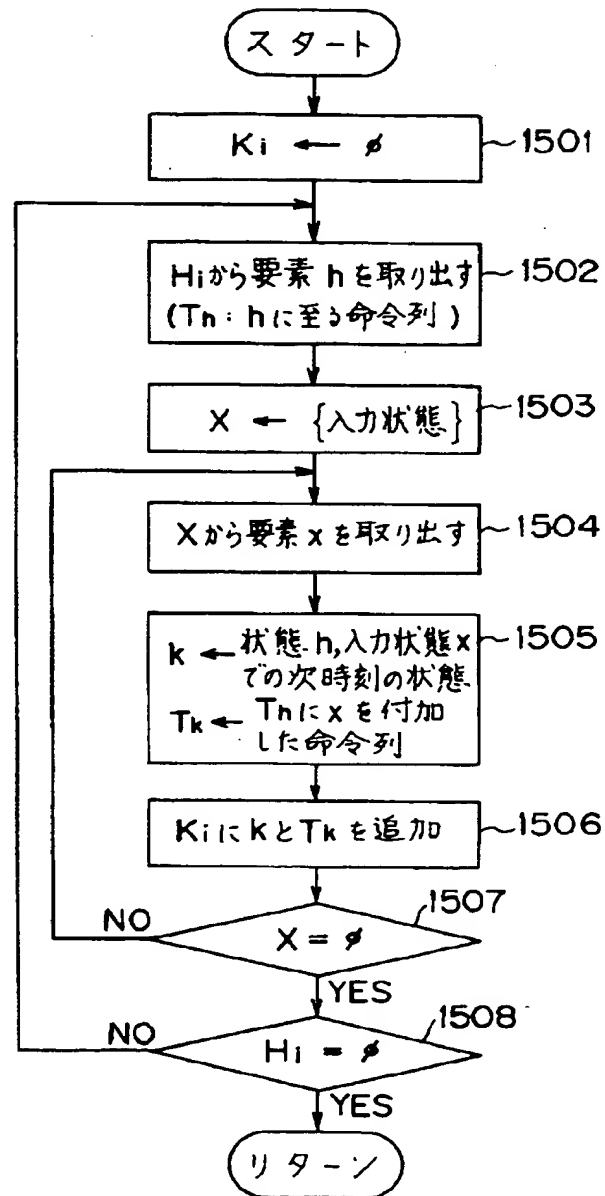
【図14】

基本遷移による次時刻の状態集合 S' の計算と
試験命令列出力の処理の動作フローチャート



【図15】

次時刻状態集合の計算処理の動作フローチャート



【図 3 2】

基本遷移の後の S' と T_0' の
データ構造の例 (その 2) を示した図

S' :	(a)	状態変数	状態	$T_0' = (A, A)$
		#1	A	
		#2	A	
		#3	O	
		#4	O	
		#5	O	
		#6	O	
		#7	O	
		#8	O	

S' :	(b)	状態変数	状態	$T_0' = (A, B)$
		#1	B	
		#2	A	
		#3	O	
		#4	O	
		#5	O	
		#6	O	
		#7	O	
		#8	O	

【図 3 3】

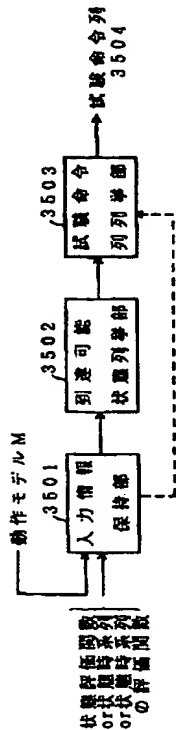
基本遷移の後の S' と T_0' の
データ構造の例 (その 3) を示した図

S' :	(a)	状態変数	状態	$T_0' = (B, A)$
		#1	A	
		#2	B	
		#3	O	
		#4	O	
		#5	O	
		#6	O	
		#7	O	
		#8	O	

S' :	(b)	状態変数	状態	$T_0' = (B, B)$
		#1	B	
		#2	B	
		#3	O	
		#4	O	
		#5	O	
		#6	O	
		#7	O	
		#8	O	

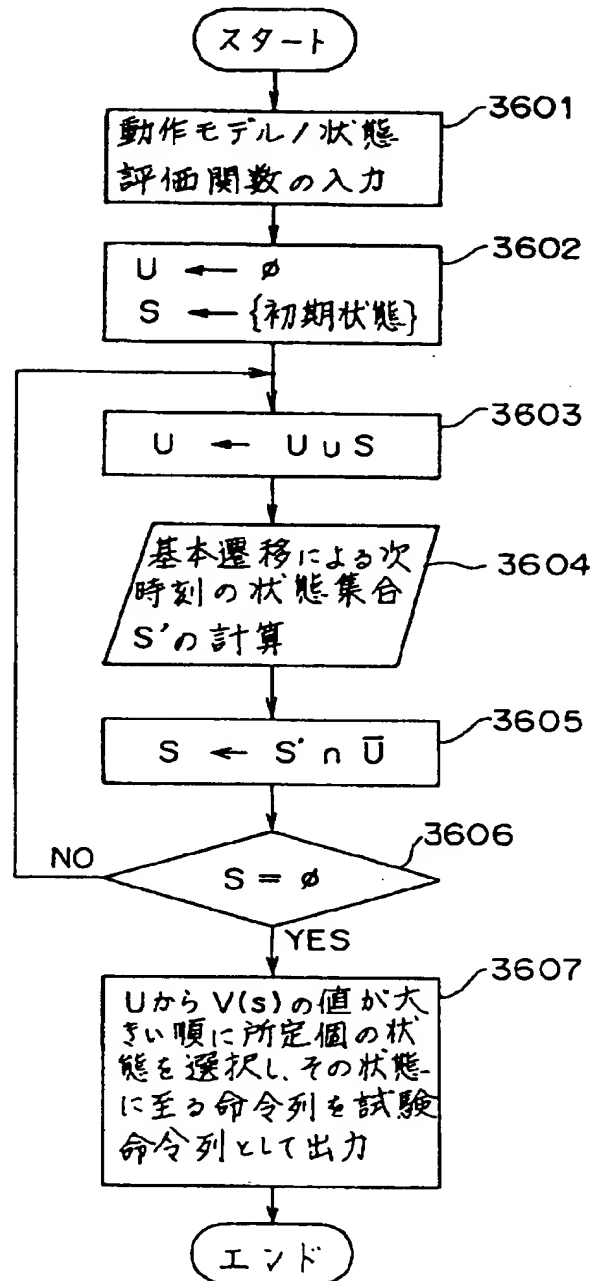
【図35】

本発明の第2～第4の実施例の全体構成図



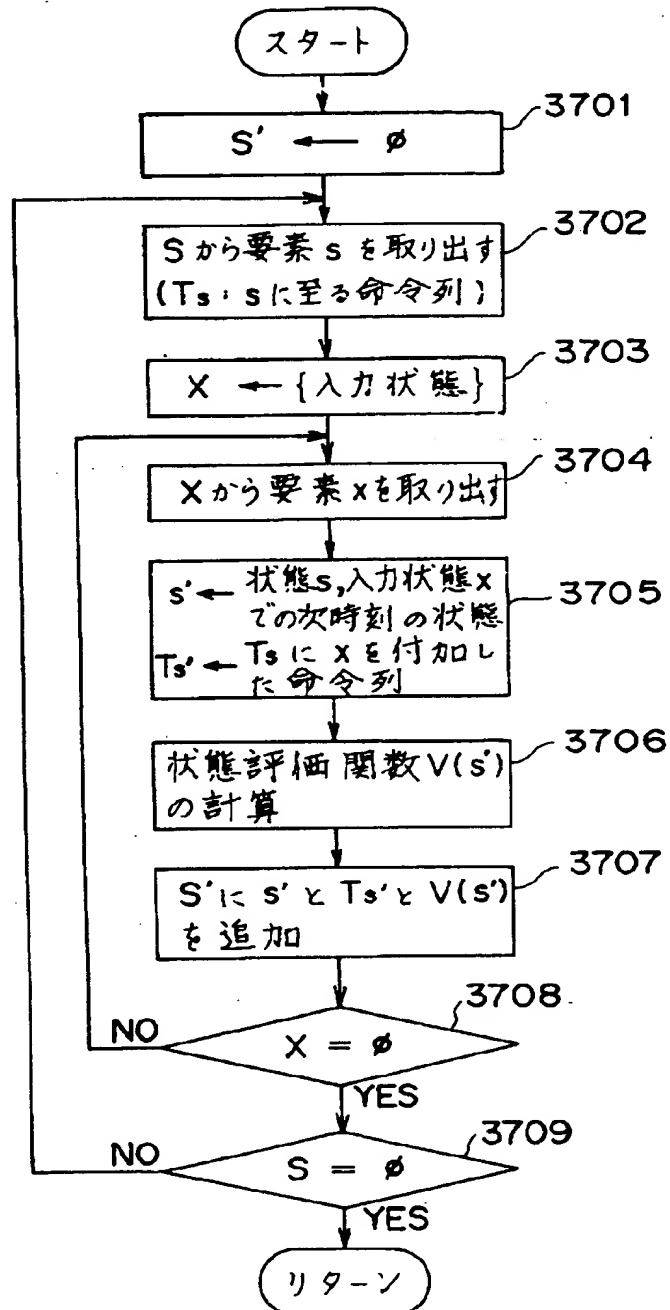
【図36】

第2の実施例の全体動作フローチャート



【図37】

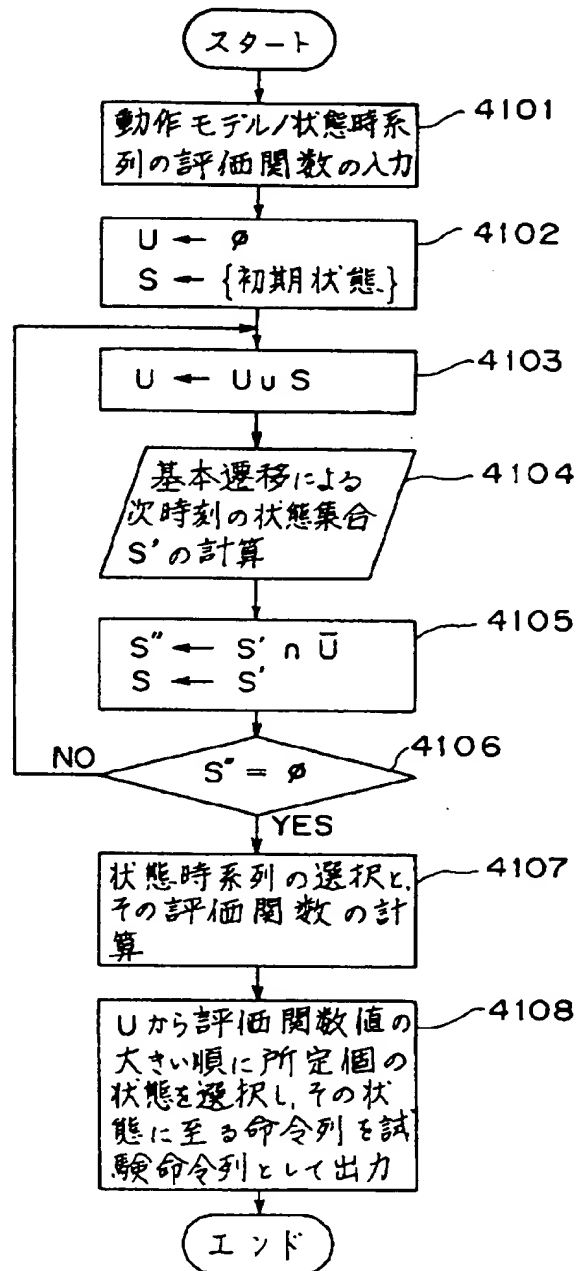
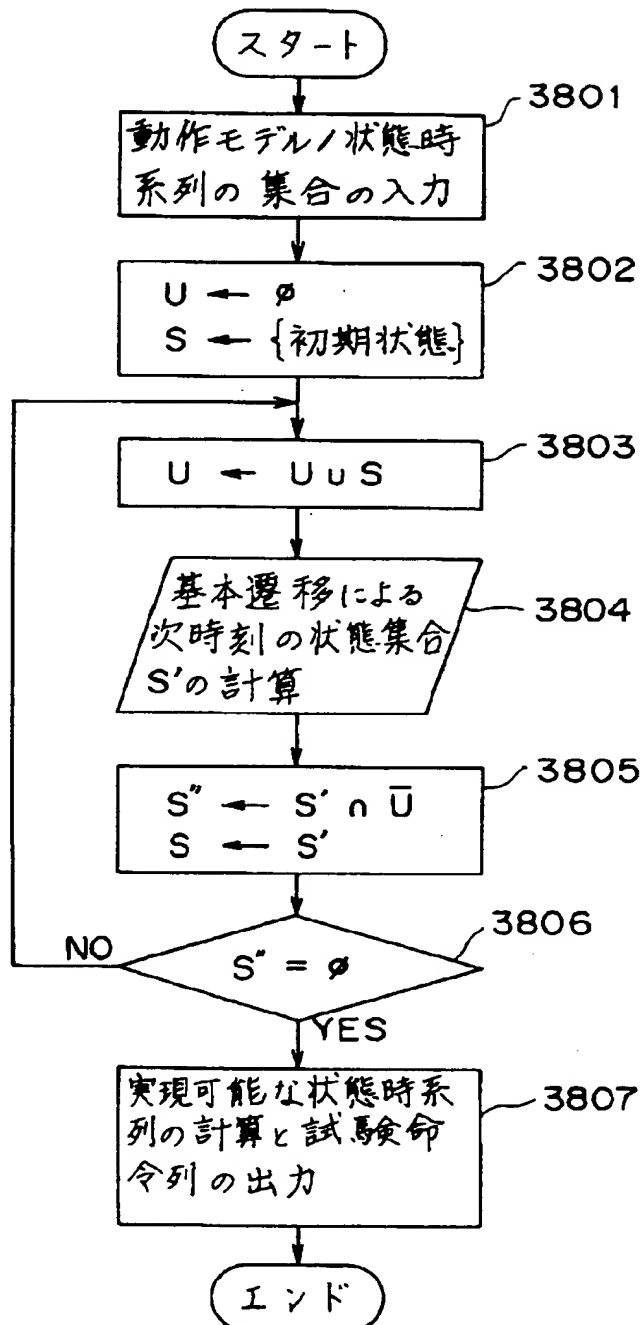
第2の実施例における基本遷移による次時刻の
状態集合 S' の計算の処理の動作フローチャート



【図38】

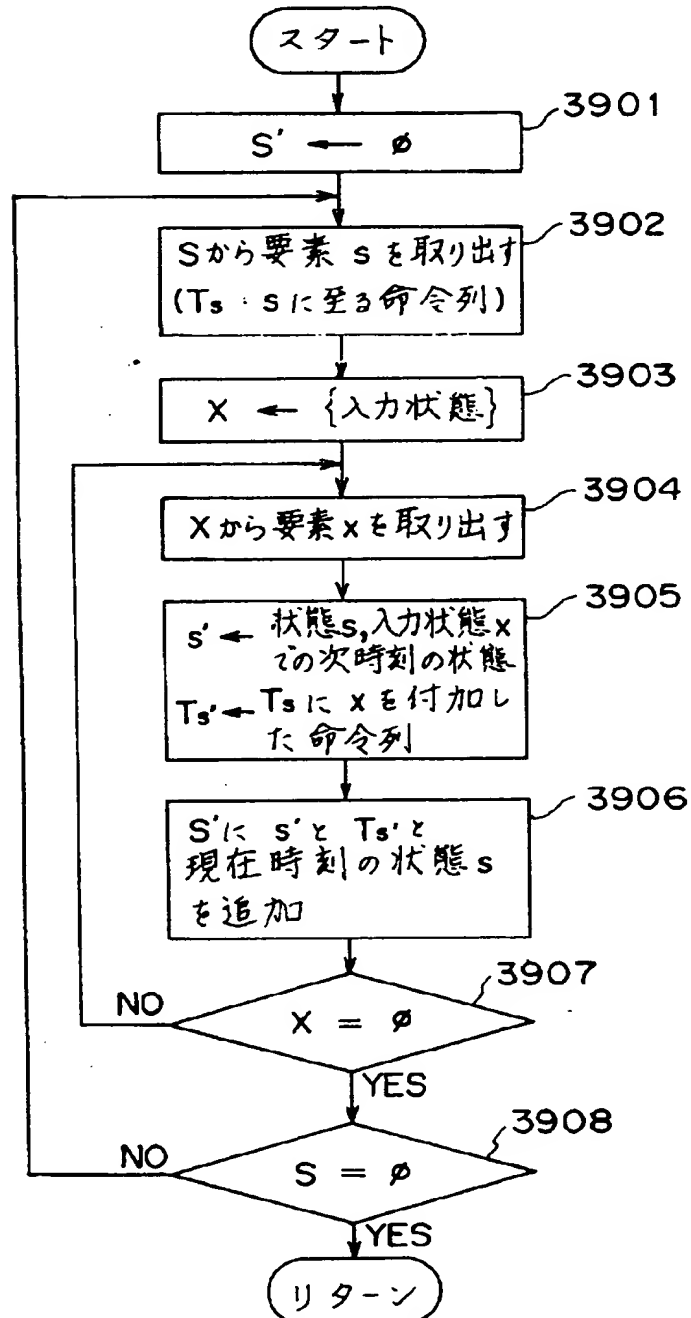
【図41】

第3の実施例の動作フローチャート 第4の実施例の動作フローチャート



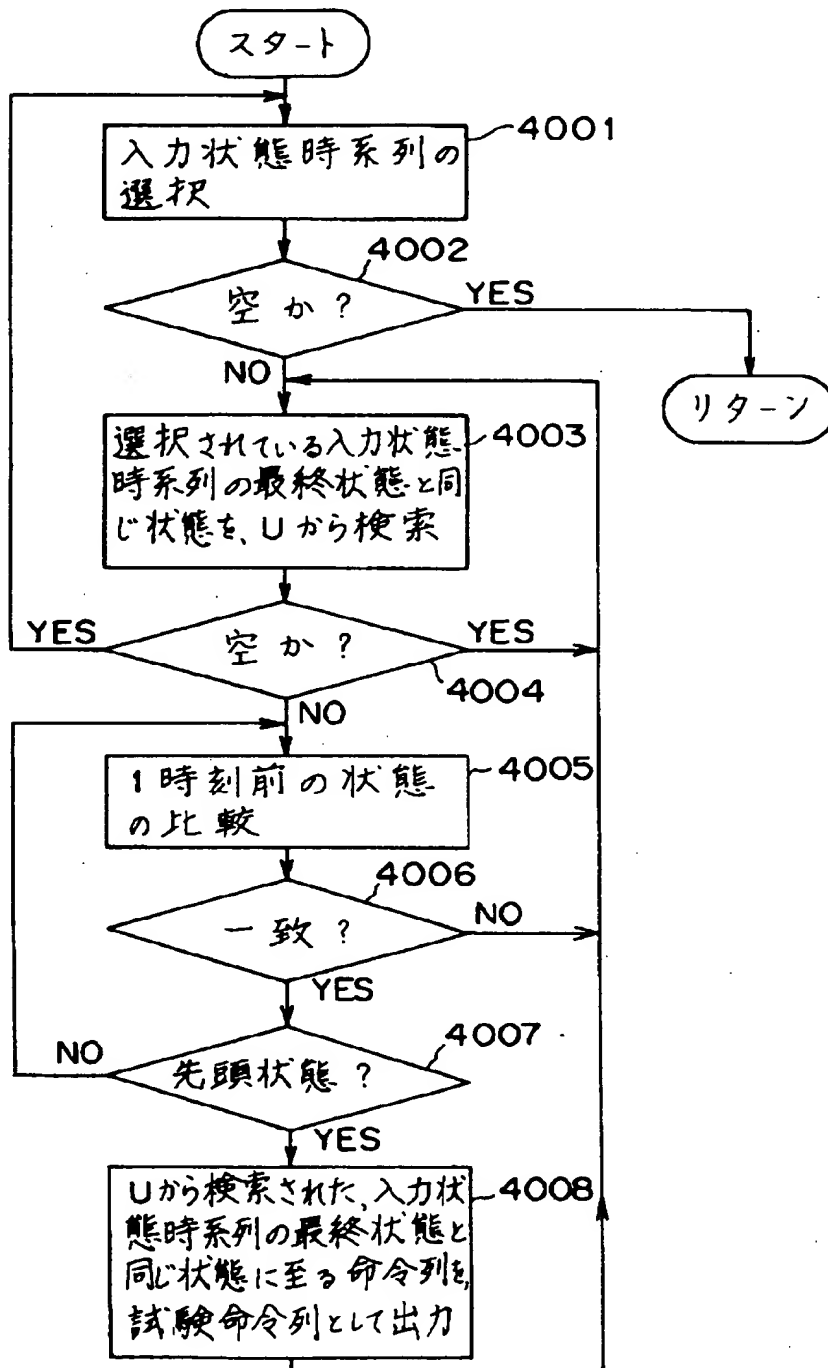
【図39】

第3及び第4の実施例における基本遷移による
次時刻の状態集合 S' の計算の処理の動作フローチャート



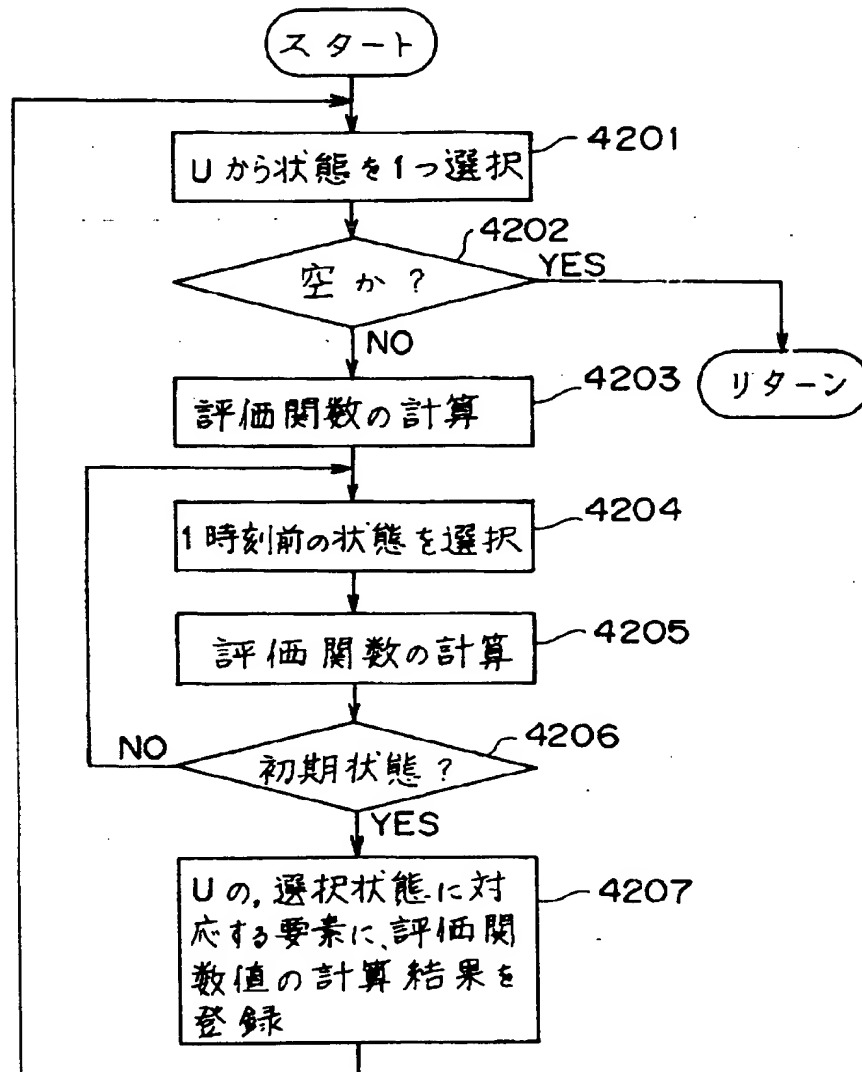
【図 40】

第3の実施例における実現可能な状態時系列の計算と
試験命令列の出力の処理の動作フローチャート



【図42】

第4の実施例における状態時系列の選択と
その評価関数の計算の処理の動作フローチャート



フロントページの続き

(72) 発明者 広瀬 文保
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内